ISSN 0956-9979 MAY 1994

VIRUS BULLETIN

THE INTERNATIONAL PUBLICATION ON COMPUTER VIRUS PREVENTION, RECOGNITION AND REMOVAL

IN THIS ISSUE:

• **Investigating Computer Crime.** UK Fraud Squad detectives start making inroads into the most puzzling 'Whodunnit' since the Great Train Robbery. Has an outbreak of computer crime swept Britain? No, it is all part of a training program. For more about *Operation Skye*, turn to page 15.

• **Pathogen on the loose.** A new virus is in the wild in the UK. It is highly polymorphic and (at this time) not detected by many vendors' software. Jim Bates dissects the latest virus 'in the wild'.

• **In Defence of the Realm.** *Dr Solomon's Anti-Virus Toolkit for DOS* is one of the best known products in the industry. Does the *NetWare* version live up to the same high standards set by its predecessors?

CONTENTS

EDITORIAL

Do Not Trust the Horse, Trojans...

In April 1994, every PC dealer's worst nightmare occurred: a Melbourne vendor discovered that a logic bomb had somehow been planted into the software which was shipped with its machines. The bomb was designed to trigger five months after the disk was formatted, overwriting the first 128 sectors of the drive and altering the disk type setting in the CMOS to zero. The event serves as a timely reminder of the threat posed by an extremely powerful (and often overlooked) threat to corporate IT Systems: the Trojan horse (see page 4).

Trojanised software has been around for many years, both as a simple method of playing practical jokes on unsuspecting colleagues, and for more damaging purposes. In the early eighties, Unix 'guru' Ken Thompson described a Unix C compiler which had been trojanised in such a way that any subsequently compiled software (including other C compilers) contained the Trojan horse. The paper, 'Reflections on Trusting Trust', asserted that no compiled software, whatever its origins, could be trusted completely.

" no compiled software, whatever its origins, could be trusted completely "

Preventing the type of attack witnessed in Melbourne is extremely difficult. Once a new PC has been scanned for viruses, it is assumed that the software supplied with it is safe. Current anti-virus software provides no protection against this type of attack - even a software-based behaviour blocker could be bypassed, as the Trojan software would gain control first.

In this recent incident, the trojanised software was part of DOS. However, *Windows* provides even more space in which the modern-day Greek can hide. There are hidden features in a multitude of different applications - how can the user ensure that none of these 'features' will involve reformatting the hard drive? *Windows* comes with its fair share of jokes (for example, the *Windows* dedication screen, accessed by an undocumented key sequence), and some people have a truly bizarre sense of humour. Double click here, hold a key there… and your hard disk is reformatted. Hahaha. Obviously, one hopes that no such destructive routines exist in *Windows*, but the user is left with few alternatives other than blind trust in the supplier.

The only possible way to ensure that a machine will act as intended is to decompile all programs on the machine to source code, and analyse its function (but what with - a trojanised debugger?). While this may have been possible with early machines bootstrapped from punchcards, any programmer worth his salt would baulk at the thought of reverse engineering the titanic complexities of *Windows*. Operating systems like *Chicago* and *Cairo* will doubtless bring more of the same.

Unfortunately, there is no easy answer to this question; the problem is one of trust. Not only can the 'nasties' be hidden in software, but also in hardware, or the CPU itself. Sooner or later, we are forced to place our confidence in something. The Dark Avenger seems to have stopped writing viruses. Maybe he has found a job developing BIOSes for someone…

The PC is such a powerful tool that it ends up being used for a number of security-related applications. PCs frequently handle company accounts, or contain confidential information such as credit card details. How difficult would it be to add software to a machine which transmitted account information to a fraudulent employee's terminal? Easy, and (potentially) highly lucrative.

Must users, therefore, cower in their beds, waiting for thousands of Greeks to come pouring out of their computers and slay them? Sadly, there is little one can do to *prevent* such an attack, but there are actions one can take to minimise the damage. In the unlikely event of a particular company being attacked, the best (albeit not foolproof) defence is a sound backup - one can only hope that the Trojan merely deletes data, rather than making nefarious small changes over a long period of time.

Finally, the standard techniques used for virus prevention also provide some protection. Use software from a standard supplier - no matter how wonderful a new printer driver looks, recognise that there is a risk associated, however small. One can do no better than to consider the words of Virgil: 'Do not trust the horse, Trojans. Whatever it is, I fear the Greeks even when they bring gifts.'

VIRUS BULLETIN **©1994 Virus Bulletin Ltd, 21 The Quadrant, Abingdon, Oxfordshire, OX14 3YS, England. Tel. +44 (0)235 555139. /94/$0.00+2.50**

NEWS

Symantec Acquires Central Point

For some time now, there have been rumours circulating in the anti-virus industry about the future of *Central Point* and *Symantec*. Finally, on 4 April 1994, *The Symantec Corporation* announced its latest victory: a definitive agreement to merge with *Central Point Software, Inc*, in a deal estimated to be worth $60 million.

Old Conquests

Although news of the merger caused a number of raised eyebrows, those studying the industry have been predicting such a pooling of resources for several months.

Central Point is just another in a series of companies which have been acquired by *Symantec* over the years. Others are:

- *Peter Norton Computing*
- *Symantec UK*
- *Certus International*
- *Fifth Generation Systems, Inc.*

The first merger brought forth *Norton Anti-Virus 1.0* (*NAV*), a sickly product which was quickly superseded. The current version of *NAV* owes many of its features to the technicians from *Certus*, and as such is a vast improvement on previous releases of the product.

Central Point itself has also acquired companies involved in the anti-virus industry. In 1993, *Central Point* merged with *XTree*, which was then selling the *AllSafe* product. The proposed merger with *Symantec* means that seven different anti-virus products will have had (albeit indirectly) some interaction with the company (*AllSafe, NOVI, NAV, CPAV, MSAV, Search and Destroy,* and *Untouchable*).

Impact

Before the merger is finalised, it must undergo a regulatory review under the *Hart-Scott-Rodino Act*, which is expected to be completed in the June quarter. Assuming that this happens, *Symantec* is set to become the sixth largest computer software company in the world. Until then, for both companies, it is business as usual.

Both companies will continue to update and develop their products independently, pending the completion of the regulatory body's study. However, *CPAV* product manager Tori Case would not be drawn on what the long term future for the *Central Point* products or the company would be: 'At the moment, the merger is subject to regulatory approval. Until that approval comes through, we will continue to function as two separate companies - users still call us for support or information on the products, and will continue to do so for the foreseeable future.'

Virus Prevalence Table - March 1994

Virus	Incidents	(%) Reports
Form	14	34.1%
New_Zealand_2	5	12.2%
Spanish_Telecom	3	7.3%
V-Sign	3	7.3%
Exebug.4	2	4.9%
JackRipper	2	4.9%
Keypress.1216	2	4.9%
A&A	1	2.4%
AMSE	1	2.4%
DIR_II	1	2.4%
Exebug.3	1	2.4%
Joshi	1	2.4%
LZR	1	2.4%
Parity_Boot.A	1	2.4%
STB	1	2.4%
Stealth_Boot.B	1	2.4%
Tequila	1	2.4%
Total	41	100.0%

Will *CPAV* continue to exist after the merger has taken effect? Case is reluctant to comment: 'We will continue to sell and develop new versions of our product. This will probably continue well after the merger happens, but until it does, we're not in a position to make any decisions.'

With any merger there is an inevitable loss of jobs. How has this news been received within *Central Point*? 'I'm not really sure whether I can comment for the rest of the staff. From a human resources point of view, it was handled very well by both companies, and I think that employees feel that they were treated fairly and honestly. Those who will be staying on are excited about what we are going to be doing,' stated Case. 'If the merger goes through, we'll be part of a much larger public company, and together, particularly in the network utilities market, we'll be able to do things which neither company could do by themselves.'

Better, by Being Bigger

What effect the merger will have on the anti-virus industry is difficult to predict. However, one thing is clear: a combined *CPAV/NAV* userbase would make *Symantec* the main player in the industry, giving it still more muscle. An unnamed source at the Washington *NCSA* conference commented, '*Symantec* appears to be setting out to have the best product in the industry by having the *only* product in the industry. Is nobody safe?' Who will be next on the *Symantec* hit list? [*Bill Gates, watch out! Ed.*] ▮

The HDZap Trojan

[*From a report submitted by Roger Riordan, of Cybec Pty.*]

When PC users in Australia returned from the Easter break and turned on their PCs, a substantial number received a nasty shock when their machines refused to boot. When the problem was investigated, it was found that the CMOS setup had been corrupted, and that the start of the hard disk appeared to have been overwritten with garbage.

Embedded Timer

Several of *Cybec's* customers were affected, and after further investigation the problem was eventually traced to a number of PCs shipped by the Melbourne firm *IPEX*. Affected PCs contained a logic bomb in their DOS Boot Sector, set to trigger five months after the PC was formatted. In addition, the DOS FORMAT program had been altered, so that it wrote the Trojan code to every disk formatted.

In affected DOS boot sectors, the normal initial jump to the boot program is replaced with a jump to a primitive, but very effective, time bomb. Each time a PC is booted, the date is read from the CMOS clock. If it is five months or more from the date on which the disk was formatted, the low byte of the timer tick counter is read. Depending on the value returned, there is a one in four probability that the first 128 sectors of the hard disk will be overwritten with garbage, and the hard disk type in the CMOS set to zero. There is another one in four chance that just the CMOS will be zeroed. If any of these conditions are not met, the program will jump to the normal boot procedure.

In the Trojan version of FORMAT.COM, the time bomb has been inserted into the boot image which is written to each disk during formatting, and the start of the program replaced with a jump. This points to a very short program segment which reads the date from the system clock, and adds five months to get the trigger date. The Trojan plugs this into the boot image, and then lets FORMAT run normally. The altered code has been written over part of the *MicroSoft* copyright notice, and the logic bomb replaces the normal disk error messages in the boot sector.

Long Term Risk

It has been established that corrupted PCs have been supplied to schools throughout Victoria, to government departments, and to at least one private firm in New South Wales. The number of PCs which have already had the contents of their hard disks destroyed probably exceeds one hundred, and is likely to increase substantially with time.

Furthermore, every floppy disk formatted with the Trojanised FORMAT program will contain the time bomb, and if anyone accidentally boots from the disk after the trigger date, there is the same probability of destroying the hard disk. It has been established that many copies of the affected FORMAT.COM have been distributed, and that in at least some schools, these have been loaded onto the utilities directory on the file server, so that nearly all disks formatted in those schools will contain the time bomb. Much of the data on the disk can be recovered after the Trojan has triggered, but users should contact a data recovery expert, rather than attempting to repair the damage themselves.

To remove the time bomb, boot from a clean DOS master disk with the same version of DOS found on the hard disk, and enter 'SYS C:'. A DOS system disk may be used, but if there is no disk copy of SYS, it can safely be run from the hard disk, using the command A:> C:\DOS\SYS A: C:.

Arie Benexra, National Service Manager for *IPEX*, was 'too busy' to fax any details of the Trojan to *Virus Bulletin*, and claimed that only 'about 3%' of *IPEX*-built PCs were Trojanised (approximately 300). He was also unwilling to supply any further information on the Trojan, its effects or its removal. It is possible that Mr Benexra will be more helpful to paying customers. Anyone who owns a recently purchased *IPEX* PC is encouraged to contact him on his direct line, Tel. +61 3 242 5010 ∎

Nordic Naughtiness

After a brief skirmish in Finland between *Data Fellows* and *Safeco* over alleged hacking of a *Safeco* BBS, it would appear that there is still confusion over what occurred.

Risto Siilasmaa, Managing Director of *Data Fellows*, states that the employee in question (henceforth referred to as 'John') was fired as soon as the charges against him were made. However, according to John, the charges were brought after an unresolved dispute about unpaid holiday compensations. Commenting on the dispute, Siilasmaa said, 'No one thought of asking the police whether they had suspicions concerning *Data Fellows*. The superintendent in charge of the investigation would have been happy to reply that at no time during the investigation had *Data Fellows* even been suspected. After all this, we saw no choice but to sue the agent for orchestrating false rumours of our involvement.'

Hannu Öhrling, Managing Director of *Safeco* strongly denies these allegations: 'As the victim of a serious crime, we have brought the hacking case into the open in order to raise general discussion about these matters in Finland. After a series of hostile actions like this, it makes one wonder about the real motives behind all actions taken.' Whatever the truth, it would appear that things are set to escalate, and it underlines some of the problems which can occur when close competitors' workers are 'headhunted' ∎

Virus Exchange - No Thank You

VB apologises to the owner and SysOps of the Milan-based *Euforia BBS* for implying that the board has any links to the computer underground (*VB*, March 1994, p.8). Access to their virus forum is only allowed after extensive validation by the SysOp, and is limited to *bona fide* researchers ∎

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 17 April 1994. 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

Alexander.2104 **CER:** This variant probably originated in Romania, just like another 1843-byte variant. Both are detected with the Dark_Avenger pattern, and it is possible that the Alexander family should be reclassified as a group of this virus.

Ash.712 **CN:** A new, encrypted variant.

```
Ash.712         E800 005D 81ED 0B01 8D9E 2A01 538A 8622 01B9 A202 3007 43E2
```

Beer.3399 **CER:** A new variant of this Russian virus.

```
Beer.3399       FA90 80FC 3B75 03E9 17FF 3D00 3D74 0F3D 023D 740A 80FC 5674
```

Blood_Sugar **CN:** A 416-byte encrypted virus, which contains the text '[C6H12O6] Blood Sugar by MnemoniX'.

```
Blood_Sugar     F381 EB23 008A 278A 5701 89F7 FCB9 0010 AC2A C400 D4AA E2F8
```

BUPT.1261 **CER:** Detected with the BUPT (formerly Traveller) pattern.

Burger **CN:** Several new insignificant variants of this overwriting virus from Burger's book have appeared recently, and are all detected with existing patterns. They are 405.F, 505.G, 505.H, 505.I, 505.J, 560.AL, 560.AM and 560.AN.

Cascade.1699.B **CR:** Almost identical to the 1699 variant, which has now been renamed to 1699.A. Detected with the Cascade.1699 pattern.

Cascade.1701.Jojo.G **CR:** Detected with the Jojo pattern.

Cascade **CR:** Three new minor variants (1701.M, 1701.O and 1704.S), all of which are detected with the Cascade (1) pattern.

Danish_Tiny.Wild_Thing.287 **CN:** Very similar to the 289-byte variant. Detected with the Wild_Thing pattern.

Dark_Avenger.1800.L **CER:** Very similar to the original 'Eddie Lives' variant, and detected with the Dark_Avenger pattern. The same applies to three other new variants, 1797, 1800.Eugen and 1813.

Datalock.828.C **CER:** Detected with the Datalock pattern.

Ein_Volk **CN:** A variable-length virus, which contains the text 'Ein volk, Ein reich, ein führer!' [*sic. Ed.*]

```
Ein_Volk        53BB 3402 03DE 4BFE C0FE CCF6 D432 C443 3007 E2F3 5BC3 BA52
```

F-Soft.590 **CN:** An unremarkable variant of this Polish virus.

```
F-Soft.590      32ED 8D97 EE01 CD21 8B95 C901 8E9D CB01 B824 25CD 210E 070E
```

Green_Caterpillar.1575.G **CER:** Detected with the Green_Caterpillar (previously 1575) search string.

HLLO.Novademo **EN:** This is an overwriting virus which was distributed in a 257897-byte file, NOVADEMO.EXE.

Japanese_Christmas.722 **CN:** Detected with the Japanese_Christmas-Cookie pattern.

Jerusalem.1808.CT.SubZero.B **CER:** Detected with the Capt_Trips pattern.

Jerusalem.1808.sUMsDos.AN **CER:** Detected with the Jerusalem-US pattern.

Jerusalem.Sunday.K **CER:** A minor variant, detected with the Sunday pattern.

Keypress.2728 **CER:** A 2728-byte virus. Awaiting analysis.

```
Keypress.2728   7405 C707 4673 F9F5 1FC3 F606 2801 0174 0D8C C005 1000 0106
```

Leprosy.664.B **EN:** This overwriting virus was distributed in a file named MISERY.COM, and contains the text 'My friend of Misery'.

```
Leprosy.664.B   8B1E 2002 53E8 0F00 5BB9 9802 BA00 01B4 40CD 21E8 0100 C3BB
```

Liberty.2857.G **CER:** Detected with the Liberty pattern.

Magnitogorsk.2560.D **CER:** Only minor differences from the A, B and C variants, with changes made to the encryption loop.

```
Magnito.2560.D  2E8B 851F 003D FF00 7414 BE43 0003 F7B9 BD09 902E 0004 2EF6
```

Murphy.1650 **CER:** Detected with the Murphy_2 pattern. Four other new Murphy variants (1659, 1752, Murphy.Delyrium.1788 and Murphy.Napalm) are detected with the HIV pattern.

November_17th **CER:** Five new variants of this family are now known. The 900.A and 1007 variants are encrypted, and the search string is taken from the decryption loop.

```
Nov_17.690      3D00 4B74 07E9 D801 59E9 CB01 5053 5152 5756 5506 1E1E 8BFA
Nov_17.706      B42A CD21 80FA 0175 24B4 19CD 213C 0472 0204 80B6 008A D0B9
Nov_17.800.B    3D75 04A8 0174 1180 FC43 740C 3D00 4B74 0FE9 2902 59E9 1602
Nov_17.900.A    E800 005B B9A8 010E 1F83 C311 8137 ???? 4343 E2F8
Nov_17.1007     EB17 900E 1FE8 0000 5EB9 E001 83C6 1190 8134 ???? 4646 E2F8
```

PCBB.1845 **CR:** This virus contains the text 'Garibaldi 1.0', and is detected with the Plaice pattern.

PS-MPC **CN, CEN, CR, CER:** There is an unusually large number of new PS-MPC variants this month: 150.B (CN), 338.A (CN), 338.B (CN), 338.C (CN), 339.A (CN), 339.B (CN), 339.C (CN), 339.D (CN), 339.E (CN), 343.A (CN), 343.B (CN), 343.C (CN), 344.B (CN), 344.C (CN), 344.D (CN), 344.E (CN), 344.F (CN), 346.B (CN), 347.A (CN), 347.B (CN), 347.C (CN), 347.D (CN), 347.E (CN), 347.F (CN), 347.G (CN), 347.H (CN), 347.I (CN), 347.J (CN), 348.B (CN), 348.C (CN), 351.A (CN), 351.B (CN), 352.B (CN), 352.C (CN), 352.D (CN), 352.E (CN), 352.F (CN), 352.G (CN), 352.H (CN), 352.I (CN), 352.J (CN), 352.K (CN), 352.L (CN), 353.A (CN), 353.B (CN), 357 (CN), 565.B (CN), 565.C (CEN), 565.D (CEN), 569.B (CEN), 569.C (CEN), 570.B (CEN), 570.C (CEN), 570.D (CEN), 572.B (CEN), 573.C (CEN), 573.D (CEN), 573.E (CEN), 573.F (CEN), 573.G (CEN), 573.H (CEN), 573.I (CEN), 574.C (CEN), 574.D (CEN), 577.C (CEN), 578.D (CEN), 578.E (CEN), 578.F (CEN), 578.G (CEN), 579.A (CEN), 579.B (CEN), 579.C (CEN), 597.B (CEN), 597.C (CEN), 597.D (CEN), 598.B (CEN), 598.C (CEN), 602.A (CEN), 602.B (CEN), 602.C (CEN), 602.D (CEN), 603.A (CEN), 603.B (CEN), 603.C (CEN), 605.B (CEN), 606.B (CEN), 606.C (CEN), 607.B (CEN), 607.C (CEN), 610.A (CEN), 610.B (CEN), 610.C (CEN), 611.C (CEN), 611.D (CEN), 611.E (CEN), 611.F (CEN), 611.G (CEN), 611.H (CEN), 611.I (CEN), 611.J (CEN), 611.K (CEN), 612.A (CEN), 612.B (CEN), 612.C (CEN), 612.D (CEN), 612.E (CEN), 615 (CEN), G2.578 (CEN), G2.Mudshark (CN, 314), Greetings (CER, 1118), Love (CR, 557), Projekt.897 (CEN), Schrunch.442 (CN), Seven_Percent.918 (CER), Silent (CR, 397), Skeleton.570 (CEN), Skeleton.616 (CEN), Skeleton.617 (CEN), Quest (CEN, 760), Sorlec.639 (CEN), Weakley (CER, 859), Z10.683 (CEN) and Z10.687 (CEN).

PSV.B **CN:** Very similar to the original, and detected with the PSV (_354) pattern.

Raubkopie.1888.B **CEN:** 1888 (COM) or 2144 (EXE) bytes long, just like the Raubkopie.1888.A variant. Detected with the Raubkopie pattern.

Seventh_Son.473 **CN:** Detected with the Seventh_Son pattern.

Shake.C **CR:** Very similar to the .A and .B variants. Detected with the Shake pattern.

Stardot.979 **ER:** Awaiting analysis.

```
Stardot.979     BB48 05C7 075C 00BB D304 C707 2A00 B43B BA48 05CD 21F7 C501
```

Sundevil **CR:** A simple 691-byte virus, '...dedicated to all that have been busted for computer hacking activities.'

```
Sundevil        5050 070E 1F8B F533 FFB9 B302 F3A4 1FBA A601 B821 25CD 210E
```

Taiwan.677 **CN:** Detected with the Taiwan-c pattern.

Taiwan.743.C **CN:** Detected with the Taiwan-2 pattern.

Timid.303 **CN:** Very similar to the 305-byte variant, and detected with the same search pattern.

Tolbuhin.626 **CR:** A new member of this family, formerly called SK.

```
Tolbuhin.626    B42A CD21 80FA 1575 1BB8 0903 BA00 00B9 0100 8D1E 0001 CD13
```

VCL.Sorlec **CN:** A 631 byte long virus, which should be detected by every scanner able to detect non-modified VCL-generated files. This virus is not directly related to the PS-MPC.Sorlec viruses, but was probably made by the same person. Other new VCL-generated viruses this month are 379, 526, Olympic.B, Angel.1681 (the older VCL.Angel has now been renamed to VCL.Angel.436).

INSIGHT

Tipping the Scales

Dr Peter Tippett is one of the best-known faces on the anti-virus scene in the USA. As Director of Security and Enterprise Products for *Symantec*, he has seen the anti-virus world evolve and change many times over, going from a small academic environment, to a multi-national industry.

Of Conception and Girdles…

Tippett is a medical doctor as well as the holder of a Ph.D. How did someone with his qualifications become involved in computer security? 'That's a long story. I completed my dissertation using a *CP/M* computer, V-100 bus... all the early stuff. I was running a foundation which did research in the Pacific, and wrote software which could do mail merges with *Wordstar*, to try to raise money for them. I wound up developing software which would help other foundations raise money. Eventually, we became well-known for solving computer problems.' With this, *Foundationware* was born.

Tippett's name, and that of his company, recurs again and again in the early history of the anti-virus industry. Back in the days of three or four viruses, Tippett had already produced his own anti-virus software: 'When Jerusalem appeared, we realised that we had much of the knowledge required to write a good anti-virus product. We stopped development on our other products, and turned our hands to this - it took only a month to write the first version.'

'We released a product which could prevent those and other yet-to-appear viruses from spreading. We started out with *Vaccine*, a product which was preventative and checksum-oriented. It did not identify viruses by name. The company was called *Foundationware*, a term which we soon learned meant bras and girdles! Its name evolved, later, to *Certus*.'

Guarding against Disaster

The early days of the industry were characterised by the misunderstanding and fear bred by viruses. Tippett had little data with which to work, and was apprehensive of the consequences of his samples being released into the wild. He recalls the first show which he demonstrated his product at with some amusement. 'There were only three or four viruses, but we feared that if they got out, they might even take out San Francisco... we didn't know what to expect. We hired a security guard, and handcuffed him to a case containing the viruses. He stood in a booth for three days while we demonstrated them to users.'

The show was a great success: 'Within days of our product's launch, we sold a 1500-unit site licence. We doubled the price, and thought we'd be in "fat city" - in fact, it was two-and-a-half years before we sold another site licence.'

Changing to Scanners

Tippett believed, then as now, that virus scanners were not the best solution to the problem. Nevertheless, he developed the technology for *NOVI*: 'McAfee was whipping everybody, and Frisk was selling a lot of products. Things which were scanner-based, rather than preventative, were really cornering the market. We finally decided to go where the market wanted, and began to develop a product which was less obtrusive, but still able to deal with unknown viruses.'

NOVI was built around known virus detection, but Tippett was determined that the product should protect the user from unknown as well as known viruses. 'We added technology which you might now call heuristics - this would basically watch things get infected, and then uninfect them. We were successful in making it work, but failed to make people believe it was possible.'

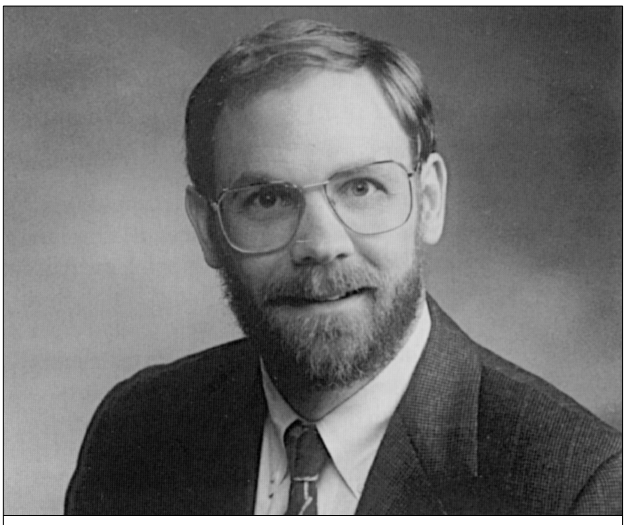
> *"there will be new viruses, ones which will require more sophisticated people and more sophisticated tools"*

'It was an interesting problem,' he elaborated, 'because the press only wanted to talk about how many viruses a scanner could find, and how many things you could get. We said that we could do that, but also had good evidence that we could detect and perfectly repair most viruses - in our tests, more than 90% - with a product which didn't require databases, checksums, etc. People didn't understand it, and we had a hard time selling it, despite the fact that it worked.'

Michelangelo Mishap

The anti-virus industry has been viciously criticised for deliberately hyping the threat posed by the Michelangelo virus. Some complaints may have been valid, but 6 March was not a boomtime for all vendors: 'For nine months after Michelangelo,' Tippett explained, 'sales were four- or five-fold less than those for the previous nine months. It was a very depressed market. *NOVI* was available before Michelangelo appeared, but our company did not have the money to launch it with a public fanfare.'

Everything accelerated at *Certus* near the now infamous 'Michelangelo Day'. Tippett recalled the energy used at that time: 'We spent one-and-a-half million dollars on marketing and promotion, and on setting up distributor channels, just when the market dropped away. Instead of sales increasing through all this marketing and so on, they decreased. We actually lost over a million dollars because of the Michelangelo incident.'



Tippett: 'Virus writing is a crazy activity. People who write viruses just don't consider the consequences of their actions.'

At about this time, *Symantec* began making overtures to the company, which led to a merger. Tippett feels that this was a good deal for both parties: '*Symantec* was looking at *MS-DOS* coming out with built-in anti-virus features, and was being beaten in corporate sales by *NOVI* - we sold to a lot of corporations for twice as much money as did *Symantec*. It was a good merger, and a lot of *NOVI* features were incorporated into *NAV*, which improved because of it.'

The change in environment from a small specialist company to a large multi-national has brought both advantages and disadvantages. Surely the product cannot react to changes in the industry as quickly as before?

Tippett agreed: 'You're right; the QA means we don't have the latest quick response to new viruses. *McAfee* and other companies will have a much quicker response in this area. On the other hand, we have a product which is tested in more depth, and more likely to work in other environments.'

In the Public Eye

Tippett's stance on computer ethics is well known, and he believes users need to explore rights and wrongs of computer use and abuse: 'We fight viruses, and people who break into systems; we fight people who steal software. However, we still haven't told people what is right, or how we should be doing things.'

One of the biggest hurdles is the lack of clearly defined moral code for Cyberspace: 'We know by the time we reach the age of twelve not to eat worms or grasshoppers. They are not dangerous to us, but we have a gut reaction that "good girls and boys" don't do that. We learn nothing from our parents about what might be good or bad in computer uses. We need to make every user start talking about ethics.'

This problem is compounded by the fact that many virus writers think that, if a virus does not destroy data, it is benign. 'Many virus writers don't see their viruses as bad,' commented Tippett - but the question of legality, in his view, is altogether a more difficult one to answer: 'Virus writing is a crazy activity. People who write viruses just don't consider the consequences of their actions. At the same time, I believe in the American constitution, and the first amendment, which gives people freedom to write, and to talk, so I don't have a problem in the larger sense of people discussing or studying viruses.'

Nevertheless, viruses *per se* clearly do pose a problem for him, in particular the authors. Tippett sees such people as morally and legally irresponsible, distributing destructive creations in such a way that they spread to other machines, without the permission and knowledge of those users.

He believes that people must be held responsible for their actions. There are, he says, legal ways to make distribution of viruses a criminal act, and believes that it is possible to define this clearly, without infringing upon legitimate needs of people either to do research or to fight the problem.

He does not see virus-writing being made illegal in the near future in the USA: 'Automobiles were around for thirty or forty years before we decided that it made sense to wear seat belts; we needed nearly two hundred years to figure out that smoking maybe wasn't such a great idea.'

'It will take a while before we, as a society, are prepared to do anything concrete about viruses. I suspect, however, that in the next few years, we will make legislation which will restrict the actions of virus authors.'

Looking Ahead

'Every time there is a shift in technology, viruses which have been active in one area cease working. Lehigh and Brain aren't seen because we don't boot from floppies anymore. They would be, if we did,' Tippett declared.

With the evolution of computer science, whole generations of viruses which are important in certain operating systems will disappear. The onward march of technology may leave current viruses behind: 'There will be new viruses, ones which will require more sophisticated people and more sophisticated tools.'

Tippett is circumspect when discussing the future of the *Symantec* product: 'I won't talk specifically about *NAV 4*, but obviously it will continue to be upgraded and improved. Our object is to create a product which works well for individual users, but which is especially good on thousands of networked machines.

'It is a fundamental part of our philosophy that end-users shouldn't have to worry about viruses, backups, their FATs - they should not even have to know what a FAT is! Security should become more automated, and users will then become free to do what they were hired to do.'

Would he give readers any clues as to what those products will be? 'Wait and see,' he replied.

VIRUS ANALYSIS 1

Pathogenic Killer

Jim Bates
Pimpernel Software

I am often asked if virus writers are good programmers. This is a little like asking if Jack the Ripper was a good surgeon - for the most part, such a question is not only irrelevant, but offensive. Any skill displayed in executing a malicious act can only make that act more abominable.

Here, the question of skill or expertise serves only to apportion the degree of guilt to be assigned to the perpetrators. If a schoolboy creates a virus without really understanding what he is doing, he is merely being irresponsible. If a skilled programmer creates a virus, he is a criminal.

I have disassembled well over one thousand viruses: many have been extremely complex, but only two or three have shown evidence of real programming skill. Hopefully, this indicates that genuinely talented programmers have too much appreciation of computing to write viruses.

The virus under examination this month, Pathogen, displays above-average programming skill, and some in-depth knowledge of anti-virus techniques. These facts, coupled with a very nasty trigger routine, make the writer one of the most destructive individuals whose work I have encountered.

Pathogen is a resident, parasitic virus which infects COM and EXE files by intercepting two DOS file access functions. The code is encrypted, and the decryptor randomised on each infection, so all appended code is changed. This method of avoiding identification has become known as polymorphism. The virus will not trigger until it has achieved a certain degree of penetration, but when it does, most of the data stored on the fixed disk will be destroyed, and the floppy drives will be disabled until the CMOS can be reset.

Going Resident

When an infected file is executed, the virus gains control and decrypts its code. A call is issued to determine if the virus is already resident. This 'Are you there?' technique is almost standard for resident viruses. Here it is achieved by placing a value of 18FFh into the AX register and issuing an Int 21h request. If the value returned in AX is 0E71h, the virus is deemed resident, and processing passes to the host program.

If the virus is not resident, processing passes to a memory allocation routine which searches the machine memory structure and locates a suitable block of memory. The virus grabs 7872 bytes from this block, which is then allocated to DOS ownership and used to store the active resident virus code. If necessary, the available memory pointer is also modified, to ensure that the virus code remains undisturbed.

The process of hooking the virus into the system begins by collecting the required system addresses from low memory, and inserting them into the virus code. Next comes a routine which traces the interrupt processing until an address within the DOS area is recognised. This address is inserted into the virus code. Tracing interrupts in this way is known as 'tunnelling', and done to try and bypass monitoring software which has itself hooked into system services. It is easy to write software which can detect and prevent this tracing process so that a good quality anti-virus monitor will not be affected by it.

This virus tunnels both the DOS Interrupt, Int 21h, and the Disk Service Interrupt, Int 13h, although only the DOS Interrupt is actually used by the virus. Once the tunnelling process is complete and the appropriate addresses collected, a counter is incremented for later testing. Next, the virus code is relocated into the prepared memory space, and the address of the virus routines are inserted into the interrupt vector table. Processing finally returns to the host program and the virus remains resident to intercept the system and disk requests.

Operation in Memory

While resident and awaiting system requests, Pathogen actively intercepts only Int 21h functions 4Bh and 6Ch (Load_and_Execute and Extended_Open/Create). Although a hook is inserted into the Disk Service Interrupt 13h, it is used in a passive manner, to reroute system requests to the tunnelled vector when they emanate from the virus code. The idea, presumably, is to avoid any monitoring activity on either interrupt when the virus code is accessing the system. The manner in which this is done suggests that the writer knows how some of the less efficient monitoring systems work and has designed this method of circumventing them.

The Int 21h handler follows a more usual pattern. The first check made on intercepted functions is whether the 'Are you there?' call has been made. This is followed by a detection routine which reroutes any exit program requests (via function 4Ch or Int 20h) to the tunnelled Int 21h vector.

The main intercepts 4Bh and 6Ch are then detected and intercepted; all others are allowed to continue into normal system services. The intercept routine is identical for both function requests, and begins by setting a flag which will indicate to other routines that the virus is in control. The virus then collects and stores the existing addresses for three other interrupts - Int 03h, Int 23h and Int 24h - and installs a default handling routine for Int 23h and Int 24h to avoid interference by DOS in the event of system errors.

The Int 03h routine is then hooked into the tunnelled Int 21h routine, to be used by the virus to access the DOS services. This is unusual, and seems to have been chosen to make

single stepping analysis of the code more difficult. It is no secret that most debugging and analysis programs use this single byte interrupt to place break point markers in code under active analysis. This is easily circumvented, but quickly becomes rather tedious for the researcher.

Once these hooks have been set, the virus checks the counter mentioned in the installation section. If the counter value is zero, processing passes to the trigger checking routine (see below). Otherwise, the infection section is entered, and the virus begins checking the target file for suitability for infection. The target file in this case is that being processed by the original system call - Pathogen does not instigate its own search for suitable files.

A Choosy Plague

The first stage of checking ensures that the target file has either a COM or an EXE extension. Then the beginning of the filename is tested to see if it begins with a pair of characters from the following list contained within the code: CO, F-, SC, TB, VI, FS, VP, VS, CL, SM and FL. If the target filename begins with any of these, it is *not* infected.

"Pathogen displays above-average programming skill, and some in-depth knowledge of anti-virus techniques"

Processing continues by collecting the attributes of the target file and storing them before resetting them to allow Read/Write access. The file is then opened, and the date and time stamps collected and stored. The date stamp of the file is checked to see if the year value is greater than 100 - if so, it is assumed to be already infected, and rejected by the virus.

If the file is acceptable thus far, its first 32 bytes are read into memory and checked for the presence of the 'MZ' or 'ZM' header bytes which identify EXE files. Processing branches depending upon the result of the test. Non-EXE files are checked to ensure that they are not greater than 56000 bytes in length - otherwise, they are rejected. A number of garbage bytes are then appended to suitable files, to increase their size to an exact multiple of 16.

EXE files are checked to ensure that they are parent programs, with no overlay number in the header field. Processing then tests the memory allocation fields within the header, and also true file size, to ensure that the file has no additional code. Next, as non-EXE files, random garbage bytes are written to make the size a multiple of 16. Extra processing is then completed to modify the EXE header segment and offset fields to ensure that the virus is processed first. A childish trick is employed to make the virus code 'appear' to be in the stack segment of the host program. This might confuse some simple debuggers and disassemblers, but is merely an empty gesture to an experienced researcher.

After the encrypted code has been appended to the host file, the attributes, date and time are returned to their original condition (with 100 added to the year value) and the file is closed. Processing then returns to the host program.

Terminal Illness

The primary check for the trigger routine has already been mentioned in the installation section. The generation counter is set such that it only increments each time the virus code is installed into memory. Thus, if the counter value is five, after installation it will become six. This will be the value in every infected copy generated during that session. There is little likelihood of the number increasing by more than two or three during the life of the virus on a specific machine.

Copies taken to other machines might also show an increase of two or three, and so on from machine to machine. The number is set to cycle from 0 to 31, and the check is looking for a value of zero, so it will only be successful after the virus has passed through several machine generations. If, for other reasons, the check fails and the virus survives, there will be a trigger-free period before the next trigger time.

However, if the primary check is successful, the system day and time is then checked. If this indicates that it is *not* Monday between 17:00 and 17:59 (inclusive), processing branches to a routine which attempts to de-install the floppy drives. Otherwise the keyboard is disabled, the screen is cleared, and a message is displayed:

```
Your hard-disk is being corrupted, courtesy of
PATHOGEN!
  Programmed in the U.K. (Yes, NOT Bulgaria!)
  [C] The Black Baron 1993-4.
    Featuring SMEG v0.1: Simulated Metamorphic
    Encryption Generator!
      'Smoke me a kipper, I`ll be back for
      breakfast.....'
        Unfortunately some of your data
       won't!!!!!
```

The virus then proceeds to write garbage at random all over the first fixed disk of the system. This process uses the low-level access routine, and will therefore corrupt any logical drives on that disk (not necessarily drive C). It is likely that the first trigger condition will occur before the second one, so it is possible that the virus will leave a system corrupted and disabled, and no external software will be able to be loaded without resetting the internal configuration.

Encryption and Infection

In viruses which use both encryption and decryption randomisation, the infection process has to be in at least two distinct stages. First, a decryption algorithm must be chosen; then a pseudo-random routine generated to achieve that algorithm. This done, the virus code can be encrypted (according to the chosen algorithm), and appended to the decryptor. In this way it is possible to produce infections of the same virus code which differ in every byte when considered in their passive state (attached to host code).

A further complication is introduced in some viruses by prepending executable garbage code. This code will execute without corrupting the programming environment, but fulfils no function other than to take up a random amount of space. Some early viruses which used this technique contained extremely complex routines to generate truly random garbage code while ensuring that memory modification was kept strictly under control.

At first sight, this virus appears to do much the same thing; closer analysis, however, reveals that there are only six different decryption algorithms, each of which has two coding methods to achieve them. Thus, identification of this virus will be considerably easier than many others, although the apparent complexity is time consuming.

Conclusion

For someone to hate computing so much that they produce a virus like Pathogen is almost beyond belief. With the added suspicion that some genuine anti-virus research was involved, the possibility of market manipulation becomes more real. Pathogen will cause problems for some anti-virus vendors due to the complex nature of the polymorphic code, and the tactic of releasing the virus straight into the wild makes the destructive trigger even more malicious.

In Singapore, I believe a person can be flogged for vandalising a car - on that scale, this virus writer would qualify for a punishment well off the Richter Scale of human endurance. The infantile ramblings in the trigger message are normal in viruses now, and the 'Black Baron' obviously fancies himself as one of the UK's top programmers. However, he doesn't even qualify for the heats, let alone for the final.

Pathogen

Aliases:	SMEG.
Type:	Resident, polymorphic, parasitic.
Infection:	COM and EXE files.
Self-recognition in Files:	Year incremented by 100.
Self-recognition in Memory:	Place 18FFh in AX and issue Intz 21h call 0E71h returned in AX.
Hex Pattern:	No hex pattern is possible.
Intercepts:	Int 21h for infection, Int 13h for stealth.
Trigger:	Disables floppy drives or disables keyboard, displays message and corrupts first fixed drive.
Removal:	Under clean system conditions, replace all infected files with known clean samples.

VIRUS ANALYSIS 2

AMSE - A Rite of Passage?

Viruses written as 'test' programs, presumably by anti-virus software producers eager to demonstrate a product, are often to be found. They are almost invariably primitive examples of basic virus technique which achieve nothing.

The advice to any purchaser of anti-virus software is: if a vendor demonstrates his product against a 'test' virus, run, don't walk, to the nearest exit - he doesn't understand the problem, so his solution will be useless.

AMSE, the virus under analysis, appears to be one such, announcing its existence each time an infected machine is booted. Its name is comprised of the first four letters of a cryptic string of characters buried within, but not accessed by, the virus code.

It is a primitive boot sector virus which infects the Master Boot Sector (MBS) of fixed and floppy disks. There is no conditional trigger routine, but it contains a rudimentary stealth capability. The code has all the appearance of a student's exercise in computing - very correct, very neat, very inefficient, and very naïve.

"the virus can survive an FDISK reconfiguration where a drive partitioning may be completely changed"

Installation

When a machine is booted from a disk (fixed or floppy), the MBS is loaded into memory, and processing jumps to the beginning of that code. On an infected machine, this is the virus code, which collects the system pointer which indicates the top of available conventional memory, moving the code up into it (reducing the pointer by around 4 KBytes so DOS will not use that memory).

Processing then transfers into the high copy of the code, and an indicator within this is tested to see which type of disk was used for the boot. The value of this indicator is used to collect the address on the disk of the virus code's remainder.

AMSE and its data occupy a total of seven sectors: on floppies, the final six sectors on the disk store the remainder of the virus code. On fixed disks, the virus uses sectors 2-7 to store its code: this will cause serious system malfunction on certain types of machine which store some vital disk type parameters on sector 2. As the virus makes no attempt to save these sectors' original contents, recovery is problematic.

After relocation of the primary portion of code, and the other six sectors read into a position preceding it, the virus collects the address of the Interrupt Service routine for INT 13h and inserts it into its own code. The virus interception routine is then hooked into the system, and a test made to see if the machine has a fixed disk. If so, this is checked, and infected, if clean. Processing passes next to a routine which displays the following message, and waits for a keypress.

```
This disk is infected by VIRUS
AMSE
Be careful !!!!
Please insert a DOS diskette into
the drive and strike any key...
```

Once a key is struck, the virus instigates a warm boot routine, leaving its code in place and active. One other area within the virus code obviously has cryptic significance to its author. There is a string of ASCII characters which do not appear to be accessed by any other section of the virus code. The fact that the first four letters are used as the virus name suggests that the remaining letters may have a hidden meaning. The letters, as they appear in the code, are:

```
AMSESLIFVASRORIMESAEP
```

Operation

When finally installed in high memory and hooked into the system, the virus intercepts only requests to READ or WRITE to sectors occupied by the virus code. If the request is to read the MBS, the virus intercepts and tests to see if the target disk is infected. If not, the virus attempts to infect it and tests again. If infected, a copy of the original MBS is collected and returned to the calling routine.

If the request is to write to the MBS, the virus reroutes this and updates its own copy of the original sector on disk. A copy is also taken of the contents of the new partition table and inserted into the infected MBS. By doing this, the virus can survive an FDISK reconfiguration where a drive partitioning may be completely changed.

If the request is to read any of the other sectors on the fixed disk used by the virus, the calling routine buffers are filled with zeros before the request is returned. If a request is made to write more than one sector of the fixed disk starting at the MBS, the virus allows the second and subsequent sectors to be written (overwriting the virus code). If the request is to write straight to the virus occupied sectors on a fixed disk, an error is returned, and the write operation is not completed.

A similar arrangement seems intended for floppy disks, but an error in the code allows such requests to continue normally. Thus, the remaining virus code can be read or written by normal BIOS system functions.

Infection

The fixed disk will be infected when a machine is booted from an infected floppy. Unprotected floppies accessed on an infected machine will be infected whenever the system reads the floppy boot sector. As this is done by the system at the first access (to determine the disk parameters), all such floppies will immediately become infected.

During the floppy disk infection routine, the boot sector is rewritten, and the remaining virus code and data is written to the last six sectors on the disk. Both copies of the FAT are altered to indicate that these final sectors are 'bad' so ordinary DOS access will not alter them.

On every disk (fixed and floppy), the sectors used by the virus are the five immediately preceding that sector occupied by the original boot code: e.g. on a fixed disk, the boot sector is in Track 0, Head 0, Sector 7. The virus code occupies Sectors 2-6 in Track 0, Head 0. This relative position also applies to every type of floppy disk.

Conclusions

This virus has all the signs of some form of test. If it was produced by a student, he earns marks of 1 out of 10 (for spelling the word virus correctly). AMSE has all the signs of a virus developed for test purpose. If this was the reason for its creation, it fails dismally, as it does nothing not already demonstrated by existing boot sector viruses. Any vendor which *must* use a virus to show its product in action, should use on of the many virus already written.

As always, if your machine becomes infected by this, or any other, virus, report it to the Police. If you see anyone using this as a demonstration virus, leave as quickly as possible and then report the details to the Police.

AMSE	
Aliases:	None known.
Type:	Master Boot Sector.
Infection:	Master Boot Sector of fixed disks, Boot Sector of floppy disks.
Self-recognition on Disks:	Virus checks 0Fh bytes of the boot sector with code stored in memory.
Self-recognition in Memory:	None.

Hex Pattern:	Located in MBS of fixed disks, or boot sector of floppies.
	`8EC0 832E 1304 04BE 007C 8BFE`
	`B900 01F3 A506 B864 50CB 061F`

Intercepts:	Int 13h for infection and stealth.
Trigger:	Displays message on each boot from an infected fixed disk.
Removal:	Disinfection possible by replacing original boot sector under clean system conditions.

VIRUS ANALYSIS 3

The Pink Panther

Eugene Kaspersky

Over the last few years, virus authors have exhausted many of the practical infection targets available on the IBM PC running *MS-DOS*, and have therefore turned their attention to hiding the presence of their creations. With a memory-resident behaviour blocker now shipped as part of the operating system, gaining direct access to the system has become of paramount importance.

The first virus to attempt unrestricted DOS access was Yankee Doodle. This virus hooks Int 01h, the Single_Step interrupt, and traces program execution when an Int 21h call is made. When control is passed to the DOS area, the virus stores the current contents of the stack, and uses this address to access the Int 21h functions during infection. Thus, any anti-virus software which monitors program behaviour will not 'see' the virus' actions. The second trick used by virus authors is to scan the DOS and BIOS areas for the code which makes up the original Int 21h handler. Unfortunately, the virus authors have not been idle: Pink Panther uses a completely new Int 21h tunnelling technique.

Installation

Pink Panther is a memory-resident parasitic EXE file infector, 4510 bytes in length. When an infected file is executed, the decryption routine restores the virus body (not completely: parts of the virus body are doubly encrypted), and passes control to the installation routine. This routine checks the DOS version number, and only continues if the system is running DOS version 5.0 or higher.

In order to ensure that the virus is not already active, an 'Are you there?' call is made. This consists of calling Int 21h with AX=3056h, BX=4D54h, CX=5A21h, DX=3933h (ASCII MTZ!0V93). The memory-resident virus returns 4F4Bh (ASCII OK) in AX register.

If this routine goes unanswered, the virus alters the Int 01h vector in such a way that it points to a second encryption/decryption loop. In an attempt to make detection of the virus more difficult, there are five blocks of code (two installation routines, two stealth routines, and the infection routine) which are stored in memory in their encrypted form. Whenever the virus needs to access one of these routines, it calls Int 01h (with the necessary values loaded into the appropriate registers) to decrypt the routine, passes control to it, and then re-encrypts it with a simple XOR algorithm.

The virus attempts to install itself into the Upper Memory Blocks if sufficient space is available (hence the need for DOS 5.0 or higher) by using standard Int 21h calls. If this memory is not free, Pink Panther loads itself into the top of conventional memory. Once resident, it attempts to obtain direct access to the DOS services via the original Int 21h handler. This is described in detail below.

Closer to the Source

The first step towards bypassing any system monitors is to use an undocumented Int 2Fh call to get the segment address of the DOS area. Pink Panther then obtains the segment address of the first occupied memory block after this (typically this will contain a device driver, loaded through CONFIG.SYS). Thus, the virus can calculate the maximum possible length of the DOS area.

The virus then allocates a block of XMS, copies the entire contents of the DOS area into it, hooks Int 06h (Undefined Opcode), and overwrites the DOS area with the word FFFFh. It then calls Int 21h, subfunction 30h (Get_DOS_Version).

"the virus authors have not been idle: Pink Panther uses a completely new Int 21h tunnelling technique"

This call is passed down through any other programs which have hooked Int 21h, until it finally reaches the DOS area. However, this memory address has been overwritten with the value FFFFh, which does not represent a valid instruction to the processor. This causes an Int 06h to be issued. The virus stores the address from where the Int 06h was generated, restores the original contents of the DOS area, and frees the allocated XMS block.

In order to understand how this gives access to the true DOS Int 21h handler, one has to consider what happens during this process. When an Int 21h call is received, control is passed from one memory-resident program to another, until it is finally passed to the DOS Int 21h handler, located in the DOS area. This code has been overwritten, and immediately causes an Int 06h to be generated. Therefore, the address returned is the address of the original Int 21h handler.

Of course, this method is too complex to be reliable, but under vanilla *MS-DOS* it works flawlessly. When using multi-tasking environments (e.g. *Windows*), things are likely to be much more complex.

Stealth Effects

Finally, the virus hooks Int 21h and Int 25h vectors and returns control to host program. Int 21h is used for file infection and stealth; Int 25h is used for stealth only.

When the virus intercepts the DOS calls Find_First_File (4Eh) and Find_Next_File (4Fh), the virus opens the file and reads its beginning. The virus checks the first two bytes for EXE stamp (the hex word MZ). If the file is an EXE file, Pink Panther makes a thorough check of the file to check whether or not it is infected. The infection markers used by the virus are as follows:

- the checksum is not equal to zero
- the entry value of CS register is equal to entry value of DS register plus one (CS=DS+0001)
- the entry value of stack pointer (SP) is less than 028Ah

If all these criteria are met, the virus assumes that the file is infected, and it returns the original length of the uninfected file. The virus also obtains the address of the System File Table of the file, and stores the address of the first sector of the file for later use.

When the DOS call Open_File (3Dh) is intercepted, the virus checks the file in a similar manner, and disinfects the file if it is infected. Thus, if the virus is memory-resident when an integrity checker is run, it will not be detected.

Whenever the calls Load_and_Execute (Int 21h, 4B00h) and Close_File (Int 21h, 3Eh) are issued, control is passed to the infection routine. Before infection takes place, the virus checks the file name against the following list of extensions: TB*.EXE, SC*.EXE, F-*.EXE, VS*.EXE, CL*.EXE, CP*.EXE [*Representing TBAV, SCAN, F_PROT, VSHIELD and VSTOP, CLEAN, CPAV. Ed.*]. If a match is found, the file is not infected. The virus ensures that the file has an EXE-type structure, and then calls the polymorphic routine. Finally, the encrypted virus code and decryption loop is stored at the end of the file.

Int 25h Stealth algorithm

The virus contains three stealth routines. The first one is called on DOS Find_First and Find_Next calls (the virus substitutes the length of file), the second routine is called on file opening (the virus disinfects the file). These routines hide the virus when infected files are accessed via standard DOS calls.

However, several virus scanners examine the disks on a sector-by-sector basis, using Int 25h calls (Absolute_Disk_Read). The virus uses its third stealth routine here: if the first sector of an infected file is accessed, the virus terminates the call, returning an error code (Data CRC Error). This routine only works if the memory-resident code has identified the target file as infected - this occurs if the scanner uses the DOS Find_First or Find_Next calls.

Polymorphic routine

The polymorphic code generated by the virus is not as complex as that created by the MtE or TPE. However, the virus author has employed a few tricks to make life more difficult for scanner manufacturers.

Firstly, the entry point of the code is not constant. Usually, there are a number of instructions which precede the decryption loop. However, one time in ten, control is passed directly into the loop.

Secondly, the decryption loop is corrupted at a randomly selected offset. This is corrected by the polymorphic code which precedes the loop.

Finally, the registers used for decryption are all loaded from the stack, using the SS and SP values added by the virus to the EXE header. This points to a data area within the virus which contains the necessary information.

The Last Page?

Since the original Pank Panther virus, a slightly larger variant has been discovered. This new virus contains the following text string:

```
   - The Pink Panther 2 (*The Last One*) - (c)
MTZ '1 Jan 1994' Italy Dedicated to Federica!
[MTZ 1994]
```

The main change made to the virus is that the stealth routine uses Int 13h, rather than Int 25h. Both viruses are related to other MTZ and MTZ.Overkill viruses.

One can only hope that the author's claim about this being 'The Last One' is true - as the number of viruses continues to climb, each new virus helps contribute to scanner saturation. Pink Panther will not be the straw that breaks the camel's back, but the load is getting heavier, day by day.

Pink Panther

Aliases:	MTZ.
Type:	Memory-resident, Parasitic file infector.
Infection:	EXE files only.
Self-recognition in Files:	Multiple. See text.
Self-recognition in Memory:	'Are you here?' call with INT 21h, AX=3056h, BX=4D54h, CX=5A21h, DX=3933h ('MTZ!V039') The memory resident virus returns 4F4Bh ('OK') in AX register.
Hex Pattern:	No search pattern is possible in files.
Intercepts:	Int 21h for infection and stealth, Int 25h for stealth, Int 01h for on-the-fly encryption. Int 06h during installation and Int 24h on file infection.
Trigger:	None.
Removal:	Under clean system conditions identify and replace infected files.

FEATURE

The Thin Blue Line

Two hackers sit conversing on a BBS, hundreds of miles apart. The only noise in the room is the hum of computer terminals and the incessant chattering of an outdated line printer. The screen of the terminal shows that they have just managed to gain access to a remote UNIX machine.

Without warning the door bursts inward, and in three simultaneous raids four men charge in, claiming to be police officers. Everything happens very quickly, and before the hackers have had a chance to think, they have been led away from the computer, caught red-handed. *New Scotland Yard's* latest raid has been a success.

Caught in the Act

The scene described above is accurate, if slightly misleading. Each 'hacker' arrested and charged was in fact involved in *Operation Skye*, the computer crime investigation course run by *New Scotland Yard's Computer Crime Unit* (*CCU*), at the *Police Staff Training College*, Bramshill. Over a four-week course, officers from many of the regional fraud squads pitted their wits against a problem which even the famous Sherlock Holmes could not have cast aside as elementary.

The Bramshill programme consists of two separate courses, tackling different aspects of the problem. During the first week, officers attending the course were instructed in how to obtain evidence from a DOS-based computer. Obviously, as computers become more commonplace, they will crop up as evidence with increasing regularity. As yet, most criminals still do not realise the evidential value of their own PCs. 'A lot of people still assume that, when you delete something, it is gone for ever,' commented Detective Sergeant Simon Janes, one of the officers responsible for running the course. 'However, you and I know that that is not the case.'

The second course lasts for three weeks, and tackles many of the different aspects of computer crime - hacking, viruses, unlawful access, telecommunications fraud etc. Although this course and others like it are designed for detectives from the regional Fraud Squads throughout England and Wales, the current course attendees ranged from far and wide, with one officer travelling from Hong Kong to attend.

Aims and Objectives

The objective of the course is very simple: that at least one member of each regional Police Service should be capable of taking on an investigation into possible offences under the UK *Computer Misuse Act* (*CMA*). 'As it stands at the moment, the *CCU* is the only dedicated unit in England - computer crime investigation at a local level is usually dealt with by the local Fraud Squads,' explained Janes.

Each police service can nominate whom they choose to send on the course. Nominees are usually taken from the ranks of Detective Constable, Sergeant or Inspector, but this is not a hard and fast rule. This present course even included a civilian, who was employed to assist the Police.

Janes makes a clear distinction between training detectives to be computer experts and training detectives in how to investigate typical computer crimes: 'We're not training detectives to be computer experts - we could never do that, and would not wish to. We aim to teach them how to manage an incident and how to investigate it. Obviously, part of that management issue is knowing when and where to go to get help.'

The question is very much one of knowing the procedures. Janes draws an analogy with the investigation of an assault. 'If a man is stabbed, he will be taken to hospital and seen by a surgeon. That surgeon would be able to supply further information, such as the size and type of the weapon used. In order to know that, you need to know a bit about what happens in a hospital. The officer investigating may not be an expert in knife wounds, but should know how to extract the necessary information from someone who is. It's a similar situation with computers.'

DI John Austen, the course organiser, agrees. 'We try to give them confidence in dealing with computer crimes that they would ordinarily have difficulty investigating. In that way, we can best serve the needs of the victim.'

Taught or Tort?

Lectures run parallel with the development of *Operation Skye*. These are aimed to tie in with what the 'hackers' are doing - for example, the lecture on *VMS* neatly coincides with the penetration of a large *VMS* installation.

The lectures are generally supplied by experts in their particular field. 'For example,' said Janes, 'yesterday Jim Bates was at Bramshill. He was here in two roles. Firstly, he gave a lecture about DOS viruses and how they worked. Secondly, one of the teams had reached a stage where they needed to know a lot more about viruses, so in the afternoon they sat down with Jim and questioned him.'

This mixture of the practical with the theoretical gives officers the benefits of both systems. The investigative side of the course is sufficiently realistic to be able to give a sense of the 'thrill of the chase', giving the often turgid theoretical aspects more immediacy. It also allows the students to put what they have learned to immediate practical use.

As the three weeks progress, the number of virus attacks and hacking offences increases. Each team receives more and more complaints, until they have gathered enough evidence

Detective Inspector John Austen plotting at the last Bramshill training course: 'You only learn when you make mistakes…' he commented wickedly.

to move in on the suspects. During the build-up to the arrest, as much realism as possible is brought into the scenario; officers even have to undergo a mock television interview in order to inform a 'concerned public' about the extent of the problem.

Misinformation

This gradual increase of pressure is masterminded by DI John Austen, leader of the *CCU*. Austen sits in the control room, surrounded by 'survivors' from previous years' courses. Together, they form a team which keeps the students busy with reports of new incidents. Various members of staff play the role of victims of the hackers, and officers get the chance to see first-hand how confusing such an investigation can become.

A lot of work goes into each part of the scenario. In the case of the virus outbreak, Austen's team has to work out how the virus was written, what it did, and how and where it spread. With this task accomplished, the next job is to give the course members just enough information to be able to work out what is going on. Balancing between making life too easy and too hard is difficult, and clues (and red herrings) are added as necessary. 'At the end of the day, we would like the students to be able to work out most or all of it,' said Janes, 'but they are going to have to work for it.'

Many of the features of the course are drawn from the personal experience Austen has gained through the *CCU*. 'You only learn when you make mistakes,' said Austen, 'and many of the exercises on the course are based on real incidents which have happened to me or my officers.'

'The thing to remember about the course is that there's nothing to pass at the end of the day. There's no examination, and candidates aren't failed. It is all about learning. The students get out of it as much as they put into it,' explained Austen.

Hot on their Heels

The proof of the pudding is very much in the eating: Austen has set up an ambitious course which attempts to cover a lot of ground in a comparatively short time. Do the students feel that their three weeks of hard slog is of any value when they return to their units?

Casting around for opinion, the general feedback on this occasion was very positive. Mark Morris (the latest addition to the *CCU*), seemed pleased with what he had learned. 'We're into the third week of the course now. It is certainly stressful, but overall, I think it has been very useful. The thematic way in which the lectures tie in with what's happening in the investigation is good, and you really find yourself getting into the scenario. I've learnt a huge amount on this course - I think everybody has.'

Several of the officers helping to run the course had also attended previously as students. Rupert Groves, from the Bedfordshire Police Force, completed the course five years ago, and found the experience invaluable. 'At the time I came on the course, I was completely computer illiterate, and found it extremely difficult. Before the course, if I had been called in to take a complaint from someone about a computer crime, I would have been floundering. Since attending the course, I have been involved in a number of *CMA* cases which have been successful. What the course did for me was give me the confidence to approach victims of, and experts on, computer crime, and understand what was happening and what needed to be done.'

Crime and Punishment

Austen is confident that the Bramshill course makes the police forces of the United Kingdom better equipped to deal with computer crime. The police are making an effort to tackle the problem - is there anything which can be done to help them? 'The computer industry has always given us tremendous support. When we've asked for anything, it has never been refused. The problem is that the public often don't recognise computer crime. They may get violations of their system, or observe unexplained occurrences, but they don't realise that they have actually been the target of criminal activity.'

The problem of not submitting a complaint is compounded by the question of whom to complain to. 'Users also don't know where to go to explain it to the police. If you were to walk into your local police station and speak to the uniformed desk sergeant, they probably wouldn't understand what you were talking about. What I would say is that anyone who has suffered a computer crime can contact us at *Scotland Yard*, or any of the Fraud Squads around the country, and there will always be an officer to deal with them.'

Happy Conclusion

The idea of gaining confidence was mentioned several times by the students, and the results are almost certainly worth the effort. Seven days after this interview was taken, the officers involved in *Operation Skye* successfully caught their targets red-handed. Hackers had better lie low for some time - contrary to popular computer underground opinion, the police can and will pursue computer criminals.

PRODUCT REVIEW 1

Norman Virus Control

Dr Keith Jackson

Norman Virus Control is a Norwegian product first developed in 1988. Both a USA and a Norwegian version exist, and this review will concentrate on the latter release. The product takes a different angle from many of its competitors, by providing a combination of a scanner and memory-resident behaviour blocker. The software includes sacrificial programs which detect infection, a program to remove viruses, a scheduler, and online help systems.

This review will attempt to measure how well each component functions under DOS and *Windows* - lack of availability precluded testing of the *OS/2* version and of network options. I use the words 'attempt to measure' with care - quite how a behaviour blocker should be tested is an interesting (and difficult) question.

Documentation

The documentation consists of a large spiral-bound manual and a ring binder. The version of the product sent for review is intended for inclusion with other *Norman* products; the binder thus has sections left empty for other documentation.

The manual covers the DOS, *Windows* and *OS/2* versions of *Norman Virus Control*. It is well-written, easy to use, and has a thorough index. However, I have always found that multi-version manuals do not really work, as they cannot discuss version-specific detail. This is also the case here: I will further elucidate later.

The software came on a 1.44 Mbyte 3.5-inch floppy disk, which excludes users with 5.25-inch or low-density 3.5-inch drives. A Rescue Disk (see below) is also included, specific to drive A and to 3.5-inch floppy disks - why the exclusivity?

Installation

Product installation is menu-driven, with onscreen help available at several points. Installation involves selecting which components to install, allowing changes to AUTOEXEC.BAT and CONFIG.SYS, and naming a subdirectory path where files should be installed. The product offers to scan the hard disk before installation, but the results are removed from the screen before they can be read: only a short summary stating 'No viruses…' is given.

Installation is almost identical under DOS and *Windows*. *Windows* seems to run the DOS installation program in a DOS box, and creates a program group containing icons for *Norman Virus Control*, the Scheduler, and the Handbook (see below). The Scheduler inserts itself into the WIN.INI setup file and is thus executed every time *Windows* is run.

Both the DOS version of the scanner and the program which cleans viruses from infected files are compressed by PKLITE, but have messages inside their code which are used to report that the program has been infected. I surmise that the checksum code has been wrapped around the outside of the compressed executable to detect unintentional alteration of the executable file(s). These programs also contain code which says 'Press any key to rebuild'. How can this work unless another (protected) image of the executable file is maintained? The manual does not explain further.

Rescue Disk

During installation, *Norman Virus Control* offers to make a Rescue Disk, and a labelled, unformatted 3.5-inch (720 Kbyte) floppy disk is duly included. It is referred to in the manual both as a Rescue Disk, and a Startup Disk. Using two names for the same thing serves only to confuse matters.

> *"at least 14 of the 50 virus test samples (28%) succeeded in becoming memory-resident"*

If the invitation to scan the hard disk is accepted, the install program offers to make a Rescue Disk. When I accepted, the floppy disk was formatted. After a while, a message stating 'BIOS Disk I/O Error' appeared, so I was forced to format the disk myself.

On restarting, scanning the hard disk again is mandatory: this slows things down tremendously, but is the only way to reach the Rescue Disk part of installation. Although the floppy was now formatted, the program blindly reformatted it, and again failed. The problem recurred with a new 720-Kbyte floppy, and a 1.44 Mbyte floppy produced the message, 'Cannot create A:\IBMBIO.COM'. At this point I gave up with the task, and moved on.

None of this, in particular the meaning of error messages, is clearly explained in the documentation. This is one aspect of using a single manual for disparate versions of a program. It seems an attractive short-cut, and looks good, but falls apart when system-specific problems arise.

Freeze - or the Canary gets it!

The main component of *Norman Virus Control* is a behaviour blocker, which tries to defeat viruses by detecting suspicious activity and asking the user whether such activity should be permitted. This is implemented as a memory-resident device driver which detects known and unknown boot sector and parasitic viruses, and is installed as the first line of CONFIG.SYS. Although it can be run in high memory, this would mean it would no longer be the first line

of CONFIG.SYS, and other (possibly harmful) software could take control first. Use of conventional memory is therefore recommended.

The device driver is aided by two programs, CANARY and S-CANARY. I have no idea what the 'S' means [*Apparently, it stands for System. Ed.*]. CANARY attempts to detect the presence of a parasitic virus by becoming infected by it. S-CANARY tries to detect boot-sector viruses, but I am not sure how - the manual says to refer to the online help, which is itself not very explicit. The CANARY programs are named after the canary of the coal mine in 'olden days', which indicated the presence of a problem by 'dying'. When a CANARY program is executed, survival without alteration equates to lack of virus infection.

The device driver monitors PC operation for 'program behaviour that represents typical virus techniques', a phrase not explained in the manual beyond the vague comment that the software is built around the 'Virus Behavioral Pattern Detection Model'. I presume this means that it monitors PC activity, seeking signs of virus activity. A generic cleaning routine for boot viruses is provided, but if used, the disk in question cannot perform a system boot (unless reformatted).

I tested how well *Norman Virus Control* detected virus activity by creating a batch file which would execute in turn each of the viruses in the test-set, execute the CANARY program, and monitor available memory to see if the virus had become memory-resident. The viruses and batch files were executed from a bootable floppy disk on a PC from which the hard disk had been removed. To ensure that the sequential execution of two different viruses did not interfere with each other, the floppy disk was remade from a master copy, and the PC rebooted, each time *Norman Virus Control* failed to prevent the virus activating.

This proved to be a slow process. The presence of a hard disk would have necessitated remaking the hard disk with FDISK, and then reformatting it, for every virus. This would have made the test procedure interminable.

```
Canary 1.55 (C) 1990-93  KWa,  Norman Data Defense Systems, Inc.

This program will check itself for any possible known or unknown
virus infection.  If Your machine is infected and You want help,
we can be reached on phone: USA (703) 573-8802   Europe +47-32-81-34-90
                      Fax:     (703) 573-3919          +47-32-81-35-10
                      BBS:     (703) 573-8990          +47-32-81-35-40

EXE: The Canary Bird Lives and all is well.
COM: The Canary Bird Lives and all is well.

c:\norman2 (19:27:09)s-canary

S-Canary v1.55 Copyright (C) 1991-93 CaB, Norman Data Defense Systems

This program checks the PC's system areas for possible infection
by known and unknown computer viruses. If your PC is infected,
and you wish to get in touch with us, we can be reached on
     Phone: USA (703) 573-8802   Europe +47-32-81-34-90
     Fax  :     (703) 573-3919          +47-32-81-35-10
     BBS  :     (703) 573-8990          +47-32-81-35-40

The S-Canary lives and all is well.

c:\norman2 (19:27:18)
```

If the 'canary' lives, the program is not infected. Death of the program suggests that help is needed!

The amount of rebuilding and rebooting necessary led me to give up after testing 50 parasitic viruses - it took the greater part of a day to do this much! The test viruses comprised one of each virus in the test-set (see Technical Details) predicated by a numeral, and one of each of the unique viruses taken in alphabetical order up to and including DOSHunter.

Detection of boot sector virus activity was simpler: running the DIR command on an infected floppy disk caused the memory-resident device driver to inspect the boot sector and check for infection. All nine boot sector viruses listed in the Technical Details were tested in this way, and all apart from Monkey (a very recent addition to the test-set) were detected as infected. This high success rate meant that testing of S-CANARY was superfluous; the memory-resident device driver had already detected all boot sector viruses except one.

Detection Rate

Measurements showed that only 60% of the 50 parasitic viruses tested were detected by the memory-resident component. When detection by CANARY was included, the rate rose to 70%. All bar one of the parasitic viruses successfully detected by the memory-resident device driver were caught when the virus was trying to infect a file. The exception was found whilst it was attempting to trace through memory.

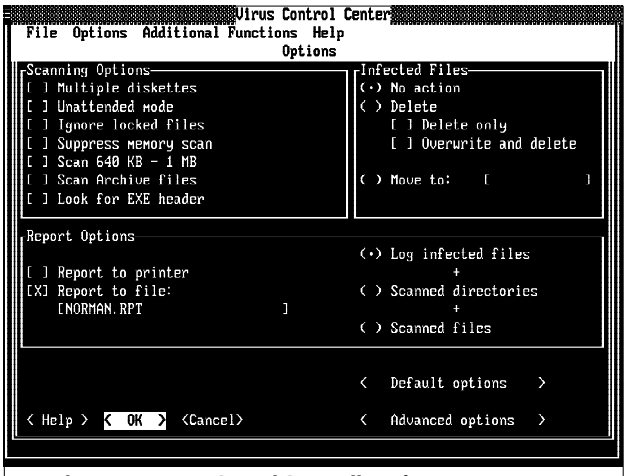
Although the memory-resident device driver claims to be able to detect when a virus makes an attempt to become memory-resident, I could see from the available memory total shown by my batch file that at least 14 of the 50 virus test samples (28%) succeeded in reducing available memory. None of the 50 samples induced any error stating that they had been prevented from becoming memory-resident.

Beyond the failure to detect attempts by parasitic viruses to become memory-resident, I am not sure what these results prove. It may be that the virus being executed inspects the system, decides that conditions are not right, and does nothing. In such a case, *Norman Virus Control* should quite rightly detect nothing. Without disassembling every virus, and figuring out exactly how each works (a task of Herculean proportions), it is impossible to verify product operation accurately. However, I would contend that, within the constraint that behaviour blockers do not work very well, the package succeeds, in general, in meeting its stated aims.

Scanning

The scanner, updated every two months, claims knowledge of 3352 virus samples. By default it scans COM, EXE, SYS, OV? and DLL files. Many options are provided, including ability to define file types to be scanned, and facility to scan most types of compressed files.

I tested scanning speed whilst the product was inspecting the hard disk of my *Toshiba* laptop (917 files spread across 33.1 Mbytes), a test which took 3 minute 20 seconds to complete.



The *Norman* Virus Control Center allows for various scanning options. Scan speed and detection rate is good, with the exception of poor MtE results.

With the *Windows* scanner, this rose to 4 minutes 21. In comparison, *Dr Solomon's AVTK* scanned the same hard disk in 1 minute 25 seconds, and *Sophos Sweep* took 2 minutes 16 seconds for a quick scan (8 minutes 58 seconds for a complete scan). This product has by no means the fastest scanner around, but its timings are quite acceptable.

The scanner detected 228 of 239 parasitic test viruses, and all nine boot sector test samples - a good overall detection rate of 96%. Noticeably, of the last update, comprising seven parasitic virus samples, it detected only one. Thus, over half of the 11 undetected test samples were related to very recently included viruses. Indeed, of the older test samples, only Jocker and Whale remained undetected.

All this implies that the scanner is being kept up to date. No doubt the missing recent samples will soon be detected. The only downside was in the Mutation Engine (MtE) detection test, where only 63 of the 1024 test samples were detected correctly - an unacceptably low detection rate of 6%.

When a scan is carried out, a detailed report is produced which may be tailored in different ways. After many viruses have been detected, the onscreen list box fills up and is replaced by a warning message referring the reader to the report file for details of viruses which have actually been detected. This is laudable, but marred by the fact that, when written to file, the report is sometimes 0 bytes long. This seems to occur when the report file already exists. So, not only is the new report not written to file, but the previous one is erased! Noticeably, the *Windows* version offers Append/Overwrite/NewName/NoReport options, none of which are available in the DOS version. All occurrences of zero length report files occurred when using the DOS version. This needs to be changed.

Miscellaneous

A utility is provided which cleans viruses from files, but in keeping with my usual stance of advising strongly against using such practices, I will not discuss the option.

The Scheduler offers a flexible system for pre-timed scans (daily, weekly, or monthly), but is only available in *Windows*. Also included are online versions of a 'Virus Library' (an overview of viruses known to *Norman Virus Control*) and a 'Virus Handbook' (a comprehensive treatise on viruses). Both are useful, and easy to use.

Conclusions

Norman Virus Control brings a clear approach to what it does. Results obtained from virus activity detection and scanning (excepting the pitiful MtE rate) are acceptable. The shortcomings discussed above could do with more explanation in the manual, and the documentation must be more specific about each individual version. Doubtless the problems encountered in making a Rescue Disk will be sorted out - this should be done immediately, as it occurs during installation, and will colour users' first impressions.

This product makes a fair job of attacking viruses using a behaviour blocker. If you want to use the scanner in its own right, my results show that it is quite acceptable in comparison with other scanners. Personally, I do not like relying on a behaviour blocker: it suffers from trying to keep up with how viruses behave, rather than what viruses look like. Finding out what viruses do will always be a more difficult problem to solve than merely detecting the presence of a virus. I'm all for simplifying difficult problems.

Technical Details

Product: *Norman Virus Control version 3.41*

Developer: *Norman Data Defense Systems*, Pb. 633 Tangen, N-3002 Drammen, Norway. Tel: +47 32 813490, Fax: +47 32 813510, BBS: +47 32 813540.

Availability: Any *IBM* PC, PS/2, PS/1 or other *IBM*-compatible system with a hard disk and one floppy disk drive (3.5-inch or 5.25-inch). One (or more) of the following operating systems are required: *PC-DOS/MS-DOS v3.1* or higher, *Windows v3.1*, *OS/2 v1.2* (text) or *v2.0* (graphics). For OS/2 and Windows systems, *IBM* and *Microsoft* recommend at least 4 Mbytes of RAM.

Price: Single user DOS/*Windows*/*OS/2* NOK 1390; Professional version DOS/*Windows*/*OS/2* NOK 2800. Prices vary for server-based versions, site licences, etc.

Hardware used: 1. An *ITT XTRA* (XT clone) with a 4.77 MHz 8086 processor, 640 Kbytes of RAM, a 3.5-inch (720 Kbyte) and a 5.25-inch (360 Kbyte) floppy disk drive, and a 32 Mbyte hard disk (a plug-in hardcard) running under *MS-DOS v3.30*.

2. A *Toshiba* 3100SX laptop PC with 16MHz 80386 processor, 5 Mbytes of RAM, a 3.5-inch (720 Kbyte) floppy disk drive, a 40 Mbyte hard disk running under *IBM's PC-DOS v6.1*.

3. A 33 MHz 486 clone with 4 Mbytes of RAM, a 3.5-inch (1.4 Mbyte) and a 5.25-inch (1.2 Mbyte) floppy disk drive, and a 120 Mbyte hard disk running under *MS-DOS v5.00*.

Viruses used for testing purposes: This suite of 158 unique viruses (according to the virus naming convention employed by *VB*), spread across 247 individual virus samples, is the current standard test set. A specific test is also made against 1024 viruses generated by the Mutation Engine (which are particularly difficult to detect with certainty).

For a complete list of viruses in the test-sets, see *Virus Bulletin*, February 1994, p.23.

PRODUCT REVIEW 2

AVTK for NetWare

Jonathan Burchill

Dr Solomon's Anti-Virus Toolkit for NetWare (AVTK) is a package which adds server-based capabilities to the DOS and *Windows* versions already available. As these toolkits have been reviewed in an earlier issue of *VB* (November 92, p.21), this review will concentrate on server aspects - how do they compare with those for *Windows* and DOS?

The *AVTK* is different in approach and structure from most other products. Its primary role is the construction of a virus defence policy; it cannot be regarded as an 'install, point, click, and forget' piece of software. It is well-constructed, avoiding unnecessary duplication of features between the workstation and server-based software, and has a powerful control language.

This is, however, not a product for the inexperienced user: correct installation and use requires considerable investment in time and effort, from someone with at least the rudiments of batch file or job-control-type programming.

Product Presentation

The *AVTK* contains copies of *Dr Solomon's AVTK* for DOS and *Windows*, along with the server version. The software requires just four write-protected 3.5-inch high-density diskettes, two for DOS and one each for the *Windows* and server code. Server protection consists solely of NLMs: this means that only servers running *NetWare 3.x* or higher can be protected. In fairness, none of the other server-based anti-virus toolkits reviewed recently have protection for older *NetWare 2.0* servers - perhaps it is time to retire those 286-based machines.

No mention is made in the manual of *NetWare 4.0* compatibility. Whilst there is no reason for the NLMs not to work on such a system, version 4.0 has features, such as automatic file compression and migration of little-used files to back up media, which should be specifically addressed.

I was also surprised to find no reference to *Apple Macintosh* viruses. Many networks have some *Mac*-based nodes and, although *Macintosh* viruses may not be as prevalent as PC viruses, such nodes are often used by people involved in publishing and presentation: these types of users tend to exchange disks with the outside world, and so represent a fairly high risk.

Documentation

The documentation supplied with the toolkit is excellent in content, and well set out. In addition to the installation and user manuals, a copy of *Dr Solomon's Virus Encyclopedia* is included. This provides a wealth of background information on the history of viruses, anti-virus product development, and specific virus actions. The manuals also contain information on options from which to choose upon discovering a virus outbreak. This is presented in a thoughtful, logical and 'Don't panic' manner. Over and above the printed encyclopedia, the *AVTK* for DOS also includes an electronic on-line version of the information mentioned.

A one-page, quick-start guide is included, intended to enable users who are desperate to type 'FindVirus' - and the product will do just that. This guide stresses, however, that if anything should be reported as a virus, further action should be taken.

"the AVTK is … an excellent server-based scanner with a flexible and configurable scheduler"

Installation

The first surprise I had when installing the software is that there is no install program! It should be pointed out that the install procedure takes barely more than a printed page to describe, and that there are only four files to be installed on the server: two NLMs, a parameter file and the virus scanner driver/database. However, doing all this by hand gives a somewhat rustic feel to the process, and certainly requires a network administrator-like mentality (which could well be desirable in what is essentially a security issue).

The install instructions recommend using the ATTACH command rather than LOGIN, in order to avoid running the login script. Loading of installed NLMs must be done from the server console, although the remote console (RCONSOLE) works just as well.

I found the lack of install program rather strange. There may not be a lot to do, but other packages manage to automate the installation, load the NLMs automatically, and give options for loading of NLMs at server boot time by modifying AUTOEXEC.NCF. Without an install and configure utility, *Dr Solomon's AVTK* cannot offer any of these features automatically, and does not even mention adding automatic startup.

The NLM Scanner

The NLM virus scanner, called NFINDVIR.NLM, will be immediately familiar to anyone who has used DOS FINDVIRU - it is merely a server-based version of the same utility. The NLM uses almost exactly the same command

line options as the DOS-based version, and works in much the same way. When a virus is detected, one of four actions will be taken:

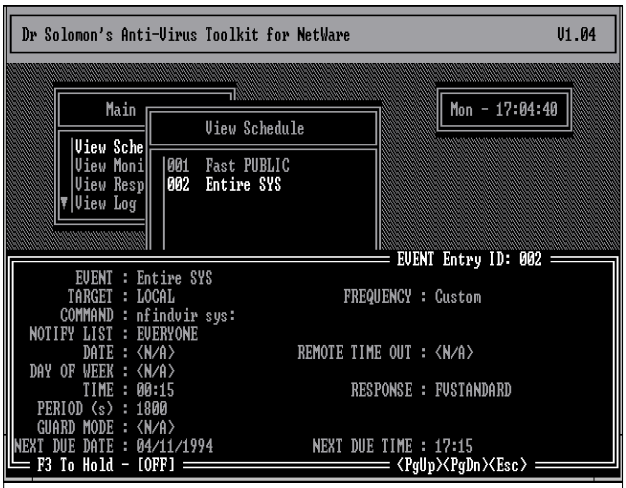
- The infected file is renamed, and file permissions altered to allow only supervisor access.
- The infected file is moved to a dirty directory, which is inaccessible to normal users.
- The infected file is backed up to a dirty directory, and an attempt is made to repair the file automatically.
- The infected file is deleted and purged.

One feature of the file moving option is that the directory structure of the source file is preserved in the confinement directory, which makes it simple to discover which source directories had infections. It also makes reviewing a snap, as all the test-sets can be scanned in a single pass!

Version 6.61 of the *AVTK* was sent for review, which uses a new detection/repair algorithm, the Generic Decryption Engine. This claims to be able to detect and repair most polymorphic infections. I did not try out the repair aspects, but found detection excellent. The scanner claims to know 3829 viruses, and is no slouch when it comes to detection: the only test-set where the product missed a significant number of files was the Polymorphic test-set, where 12 samples of the Cruncher virus went undetected.

The scanner is designed to be used in a batch mode. No option is offered to scan files automatically, as they are written or read from the file server; instead, this protection is provided by the workstation utility called VirusGuard. This seems to be a poorer alternative to on-line scanning by the NLM, as it takes up valuable memory and processor time, two commodities in increasingly short supply.

The scanner is command-line driven, and can only be started by manual commands issued at the system console (or via RCONSOLE), or by being spawned from another NLM



Pretty it certainly isn't, but the *Anti-Virus Toolkit for NetWare* is a quick and accurate utility, capable of detecting a large number of viruses and Trojans

(such as the scheduler). There is no user interface to the scanner at the console, and there are no workstation utilities to start and monitor scanner progress.

Options allow for reports to be sent to a disk file and/or the printer. The scanning overhead was fairly minimal on the test server; an optional-SLOW parameter allows a delay period in milliseconds to be specified between each file inspection. This can be used to reduce the scanning overhead further, at the expense of increased scan time. Rather than specifying all the choices on the command line options, these may be placed in a server-based parameter file, which will be read automatically on scanner startup.

All possible result codes from the scanner are thoroughly documented. As no utilities are provided to scan and write reports from the log file, it may well be necessary to use the documentation to construct one's own report generator.

The NLM Scheduler

The scheduler coordinates scanning activity across the network, as well as scheduling scanning activity and communicating with the workstation TSR, VirusGuard. It contains the connection monitor, the volume mount monitor, and the workstation alert monitor, all three of which watch and track various types of system activity. The connection monitor tracks workstations as each one logs on to the server, enabling the scheduler to perform workstation checks at login time. The Volume mount monitor tracks *NetWare* volumes as they are mounted, and allows automatic scanning of a volume at mount time. The workstation alert monitor tracks and logs alerts received from the workstation.

The scheduler is controlled by an ASCII file called NTOOLKIT.INI, which must be in the same directory as the scheduler. The scheduler is programmed via a job control language, which makes it extremely flexible, but requires a fair amount of learning to be undertaken. Absolutely no utilities are provided to edit and configure the control file. I fear it is out with the ASCII editor, gird the loins, and put up the safety net. By way of example, the following script automatically scans a volume at mount time:

```
MOUNT(TITLE=Volume mount;
     WHEN=ANYVOLMOUNT;
     COMMAND=nfindvir %VOLID:/
REPORT=%VOLID.REP;
     RESPONSE=FVSTANDARD;
     NOTIFY=EVERYONE)
```

The only real user interface in the whole of the AV server toolkit is the console screen of the scheduler (this is accessible by the workstation only via RCONSOLE) This allows viewing, but not editing, of the current schedule control file. This is in terms of what action will take place at which time, and gives some confidence that the control file has been correctly programmed.

Other options available include viewing the workstation alerts, the log file and the current schedule control file, and clearing active events from the scheduler.

Communications between the scheduler and VirusGuard use encrypted packets. This is presumably meant to prevent someone with a network monitor from discovering the current defence strategy, or to stop anyone attempting to subvert the scheduler with false messages.

VirusGuard

Workstation protection and scanning of files going to and from the file server is provided by the DOS TSR GUARD.COM. This is the same as the facility in the DOS Toolkit, but with an extra switch to enable network communications. The Scheduler NLM can be configured so that when a user logs on, it checks for the presence of the TSR on the workstation. If it is absent, login can be optionally prevented until it is installed.

VirusGuard, as the server software, has no install program - it must be manually installed. It keeps its size in conventional memory small by accessing a signature data file either from disk, or from XMS or EMS RAM. A helpful section of the manual deals with problems which may be encountered with diskless workstations.

The performance of VirusGuard was good on the Standard and In the Wild test-set. However, when used on a subset of the polymorphic test-set, it missed all samples of Coffeeshop and Uruguay.4, only identifying *some* of the Cruncher infections. Given that VirusGuard provides the on-access file scanning for the system, these results are a cause for concern. Coffeeshop is known to be in the wild - should a virus slip through the TSR protection, it could spread undetected until the next scheduled scan.

Conclusions

The *AVTK* is just what it claims to be: an excellent server-based scanner with a flexible and configurable scheduler. However, the protection provided for the workstations and the lack of effective on-access file-scanning is disappointing. Additionally, it is a toolkit with almost no user interface and no workstation utilities for access and configuration - consultants, programmers and hackers will love it! System administrators may well be put off by having to learn the control language. Such people might thus use it in occasional 'basic scan' mode, which would compromise security.

There is no concept of grouping servers into domains (due to the lack of scheduler-to-scheduler connectivity options), so configuration and installation has to be carried out for each server individually. In a large organization with many servers, this can represent a considerable overhead.

When compared to other server-based anti-virus products, the configuration issues must be considered. However, I suspect that the Toolkit is more versatile than some of the competition. The main scanning engine is fast and adaptable, and as such provides a good way of protecting a file-server, but requires more configuration examples for use by the less technically minded.

Anti-Virus Toolkit for NetWare

Detection Results

NLM scanner

Standard Test-Set [1]	227/229	99.1%
In the Wild Test-Set [2]	109/109	100.0%
Polymorphic Test-Set [3]	433/450	96.2%

DOS Scanner

Standard Test-Set [1]	227/229	99.1%
In the Wild Test-Set [2]	109/109	100.0%
Polymorphic Test-Set [3]	433/450	96.2%

Scanning Speed

Speed results for an NLM product are inappropriate, due to the multi-tasking nature of the operating system. Full comparative speed results and overheads for all current NLMs will be printed in a forthcoming *VB* review.

Technical Details

Product: *Dr Solomon's Anti-Virus Toolkit for NetWare, v6.61*

Manufacturer: *S&S International*, Berkley Court, Mill Street, Berkhamsted, Hertfordshire HP4 2HB, UK. Tel +44 442 877877, Fax +44 442 877882.

Price: Single server price - £399.00 with quarterly upgrades, £599.00 with monthly upgrades.

Hardware used: Client machine - 33 MHz 486, 200 Meg IDE drive, 16 Mbyte RAM. File server - 33 MHz 486, EISA bus, 32 bit caching disk controller, *NetWare 3.11*, 16 Mybte RAM.

Each test-set contains genuine infections (in both COM and EXE format where appropriate) of the following viruses:

[1] **Standard Test-Set:** As printed in *VB*, February 1994, p 23 (file infectors only).

[2] **In the Wild Test-Set:** 4K (Frodo.Frodo.A), Barrotes.1310.A, BFD-451, Butterfly, Captain_Trips, Cascade.1701, Cascade.1704, CMOS1-T1, CMOS1-T1, Coffeeshop, Dark_Avenger.1800.A, Dark_Avenger.2100.DI.A, Dark_Avenger.Father, Datalock.920.A, Dir-II.A, DOSHunter, Eddie-2.A, Fax_Free.Topo, Fichv.2.1, Flip.2153.E, Green_Caterpillar.1575.A, Halloechen.A, Helloween.1376, Hidenowt, HLLC.Even_Beeper.A, Jerusalem.1808.Standard, Jerusalem.Anticad, Jerusalem.PcVrsDs, Jerusalem.Zerotime.Australian.A, Keypress.1232.A, Liberty.2857.D, Maltese_Amoeba, Necros, No_Frills.843, No_Frills.Dudley, Nomenklatura, Nothing, Nov_17th.855.A, Npox.963.A, Old_Yankee.1, Old_Yankee.2, Pitch, Piter.A, Power_Pump.1, Revenge, Screaming_Fist.II.696, Satanbug, SBC, Sibel_Sheep, Spanish_Telecom, Spanz, Starship, SVC.3103.A, Syslock.Macho, Tequila, Todor, Tremor (5), Vacsina.Penza.700, Vacsina.TP.5.A, Vienna.627.A, Vienna.648.A, Vienna.W-13.534.A, Vienna.W-13.507.B, Virdem.1336.English, Warrior, Whale, XPEH.4928

[3] **Polymorphic Test-Set:** The test-set consists of 450 genuine samples of: Coffeeshop (375), Cruncher (25), Uruguay.4 (50).

CONFERENCE REPORT

IVPC '94 - Alive and Well in the USA

The month of March is usually set aside for the annual 'Ides of March' conference held in the United States of America. However, after the problems experienced last year (see *VB* April 94, p.3.), *NCSA* decided to provide both users and vendors with an alternative. Thus, the *International Virus Prevention and information security Conference* (*IVPC*) was born.

The conference was held at the *Stouffer Concourse Hotel*, in Crystal City. This is barely a handful of miles from the Pentagon and other government agencies, and guaranteed a good attendance. Before the conference proper started, two courses were held: a single day 'PC Virus Tutorial', led by Robert Jacobson, and a three day 'Information Systems Security Course' given by Dr Mich Kabay. Kabay's course was designed to alert MIS Managers to the dangers to which their Information Systems were exposed, and allow a meaningful risk assessment to be carried out, followed by construction of a series of contingency plans.

Four Streams

The main conference began on 31 March 1994, with an opening address by none other than Florida-based renegade Winn Schwartau (described in the sleeve notes of his new book, *Information Warefare*, as 'the Nathan Hale of the computer security industry'). He alerted delegates to the need for legislation on various computer-specific issues. With the United States' proposed Data Superhighway hanging above users' heads, what value is there, he asked, on information?

This was the conference's only group address. Thereafter, the two-day program was split into four streams: Information Security, Virus Management, Virus Technical and Telecommunications. Attendance at *IVPC* was good, and the 110 delegates split evenly enough to make most sessions' questions and answer sections reasonably lively.

The Virus Management track was aimed at those users who wanted to gain an overview of the current situation, covering such subjects as a general introduction to viruses, how to create an anti-virus policy, and other management issues.

Not surprisingly, the Virus Technical track was of more interest to the seasoned anti-virus researcher. Even here, many old-chestnut themes were wheeled out: Charles Rutstein discussed the problems of evaluating anti-virus software (a well-presented talk, but a hackneyed subject), and the concepts of Polymorphism and Stealth once again raised their ugly heads.

Perhaps the best talk at the conference was given by Joe Wells of *Symantec*, who presented his findings from the 'Wildlist'. Many denizens of the *Internet* newsgroup *Virus-L* will doubtless have encountered the Wildlist already (see *VB* December 93, p.4), but for anyone who has somehow missed this titbit of information, Wells has been busy gathering statistics on those viruses which are known to be in the wild around the world.

Although the number of known viruses is now well above three thousand, Wells has found that the number of viruses reported in the wild more than once is less than one hundred - some comfort for those who feel that the support issues are becoming insurmountable.

"America still seems to be determined to fight computer crime with a minimum of one hand tied behind its back"

Other items of interest included a busy exhibition, featuring representatives from many of the familiar names in the industry, and an open discussion forum on computer viruses. One of the items raised during the debate was what should be done about Mark Ludwig's virus writing competition. Once again, the concept of intellectual freedom and the first amendment arose (much to Dr Alan Solomon's glee!); America still seems to be determined to fight computer crime with a minimum of one hand tied behind its back.

Worth the Trip

One of the problems faced by conference organisers is finding new and innovative material for presentations. The virus field is one in which, although the number of new viruses is climbing rapidly, most of the precautions and procedures [*and presentations! Ed.*] remain unchanged from year to year.

Making the virus stream of a conference of interest to a wide range of delegates is difficult. *IVPC* suffered from this malaise: users who have attended many (any?) of the security conferences will already be familier with much of the ground covered.

This criticism in no way means that the conference was a washout: *IVPC* was possibly the best *NCSA* conference yet, and looks set to become *the* American conference on computer viruses. For those who regularly attend such events, it provides a chance to meet and debate with some of the best known researchers in the industry, and for those who have never attended a conference, it provides a good introduction to many aspects of data security.

SUBSCRIPTION RATES

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

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

Editorial enquiries, subscription enquiries, orders and payments:

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

Tel. 0235 555139, International Tel. +44 235 555139
Fax 0235 559935, International Fax +44 235 559935

US subscriptions only:

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

Tel. +1 203 431 8720, Fax +1 203 431 8165

END NOTES AND NEWS

ThunderBYTE, the anti-virus package produced by *ESaSS*, has been **'certified' by the *NCSA***. The latest version, with full *Windows* version of the *TBAV* utilities, was due to be released at the end of April. For more details, contact *ESaSS* on +31 80 78 78 81, fax +31 80 78 91 86.

A **Live Virus Workshop** will be held by *S&S International* on 16-17 May 1994 at the Ashbridge Management College, Berkhamsted, Herts, UK. Tel. +1 (0)442 877877. Fax +1 (0)442 877882.

RG Software Systems has announced the launch of the *Vi-Spy Universal NIM*. The *NIM* (*Network Installable Module*) consists of software which runs on the client machine, even though it is stored and updated from the server. Glath is confident in the new technology: 'Even though the software is installed on the server, the work is being done at each workstation, for those actions taken at the workstation. Thus one user's actions do not cause a degradation of performance for all other users.' Tel. +1 602 423 8000.

Sophos is holding two **Computer Virus Workshops** on 18/19 May and 27/28 July, at the *Sophos* training suite in Abingdon, near Oxford. Cost for one day is £295+VAT, and for both days £545+VAT. For further information, contact Karen Richardson. Tel. +44 (0)235 559933.

Euromoney is sponsoring two seminars on *NetWare* security: one for *3.x* on 26-28 October 1994, and one for *NetWare 4.x* from 16-18 November 1994. All seminars also available on-site. Contact *Euromoney* for more information. Tel. +44 (0)71 779 8526. Fax +44 (0)71 779 8820.

The exciting *Eleventh World Conference on Computer Security, Audit and Control* (*Compsec 94*) will be held in London from 12-14 October 1994. Further details form Karen Giles at *Elsevier Advanced Technology*. Tel. +44 (0)865 512242. Fax +44 (0)865 310981.

The *VB 94* Conference will be held on 8-9 September 1994, at the Hôtel de France, Jersey. Tel. +44 (0)235 531889.

Two *AT&T Bell Laboratories* researchers who helped break a hacker case, William Cheswick and Steven Bellovin, reveal their story in the book *Firewalls and Internet Security*. Topics covered include threat evaluation, security advice, and a first-hand account of the 'Berford' hacker, who infiltrated computer networks worldwide in 1991. The book is to be published in May by *Addison-Wesley Publishing Company*, Reading, MA, USA.

Coming to a PC near you soon… Can a virus ever be beneficial? According to a report in the *Nikkei Weekly*, the answer is a definite 'Yes'. A group from the Tokyo Institute has developed a new 'virus', which is designed to travel though the computers on a network, collect information, spot glitches and report back to the network manager.

Problems on the *Internet* continue. US student David LaMacchia has been charged with illegally conspiring to supply pirated copies of major software packages world-wide on a BBS running on a *Massachusetts Institute of Technology* (*MIT*) computer system. According to a report in *Computer Weekly*, over a 16-hour period, over 150 users downloaded files from the board. Many of those using the system used the anonymous forwarding service based in Finland to hide their identities. No court date has been set for the case, but if found guilty, LaMacchia could face up to five years imprisonment plus a US$250,000 fine.

A new *Macintosh* virus has been discovered. The sample, named INIT-29-B, alters applications, system files and documents. At present, few reports of the virus have been received, but the full extent of the problem is not yet known. Users are advised to update current anti-virus software.

Mark Ludwig, author of *The Little Black Book of Computer Viruses*, has announced his latest 'get rich quick' scheme. This time the Tucson-based vanguard of computing freedom has pushed the back the frontiers of ignorance by **publishing a CD-ROM which he claims contains 'thousands' of fully functioning viruses**. *VB* will obtain a copy of the CD to ascertain the truth behind this statement.