ISSN 0956-9979 AUGUST 1994

VIRUS BULLETIN

THE INTERNATIONAL PUBLICATION ON COMPUTER VIRUS PREVENTION, RECOGNITION AND REMOVAL

IN THIS ISSUE:

• ***CARO* explained.** Many people have no idea what *CARO* is, what it does, or how to join it. Everything you ever wanted to know about the organisation but were afraid to ask is on pp.8-9.

• **The professional's professional.** For the first time ever, *VB* reviews *AVP*, one of Russia's leading virus scanners: have researchers behind the former Iron Curtain made as many advances in their work as those in the West?

• **The *Lotus* position.** *Lotus* has adopted a novel approach to the prevention of large-scale virus outbreaks within the company. How effective is its technique, and is it a practical alternative in other companies? See p.15.

CONTENTS

EDITORIAL

Better, By Being Bigger?

Now that *Central Point* is hanging safely on the *Symantec* trophy-room wall, the company has begun to decide upon the future of its latest acquisition. Judging from rumblings from the *Symantec* PR machine, it would appear that both *NAV* and *CPAV* have a future, in the short term at least. However, with the pain of merging already beginning to fade to a bad memory, the release of an all-singing, all-dancing combined product seems inevitable. Although the name of this beast is still a twinkle in the Ad-man's eye (*NAVPAV*?), if the company's *Meisterplan* is fulfilled, the offspring of the merger will ship to a vast number of sites. More importantly, if *Symantec* has inherited any claim to the upkeep and development of *Microsoft Anti-Virus* (*MSAV*), the number of users will climb still higher.

"history would seem to indicate that the products which are always kept right up to date are designed by a small, flexible team"

This expansion brings with it its own special problems. One of the concerns is that as the company grows, product development, quality assurance, design and research will begin to become an increasingly cumbersome process. Although it would be naïve to assume that all staff from the anti-virus departments of both *Central Point* and *Symantec* will be involved in the development of a new anti-virus product, the number of staff involved is likely to swell. In a small company, changes can be made on a director's whim. However, in a lumbering corporate behemoth, how many meetings, papers and committees are required to make fundamental changes to the product?

The results of last month's *VB* Comparative Review tend to support this viewpoint: the majority of the most accurate products come from smaller, specialist companies. Indeed, history would seem to indicate that the products which are always kept right up to date are designed by a small, flexible team - often just a handful of experts. Although *Symantec's* product seems to be gaining ground rapidly after the acquisition of *Certus*, the company should beware of becoming becalmed in a treacle-like sea of red tape.

This 'inertia' or resistance to change is not entirely a bad thing: if too much of a product's efficiency depends on the ability of one man, any illness or malaise could cause the product to lose ground overnight. In a large company, that dependency is reduced. Additionally, by having a very large development cycle, the finished product should be more robust and stable. Whether this will be the case for *CPANAV* (or whatever its initials will be) remains to be seen; having many cooks is no certain recipe for a good product.

As always, there is a degree of industry speculation about whether the trend of company buy-outs will continue. If it does, there will be two principal effects. Firstly, users will be presented with less choice, as the number of competitors diminishes. It is worth noting that due to the vast amount of work involved in developing a scanner *ab initio*, it is unlikely that there will be many (or any) new players coming on to the field. Secondly, when choosing a product, the buyer will have to think hard about the future of the vendor: one is purchasing a year's worth of updates and support, not a one-off 'WYSIWYG' piece of software.

The likelihood of a product becoming overly dominant is probably limited by the nature of the threat: as it becomes more popular, it will become more widely targeted. Such direct attacks are far from unprecedented: in the case of *MSAV*, the TSR component of the software was targeted by the Tremor virus before the product was even released! At the current time, the large number of different anti-virus products on the market means that the targeting of any one is the exception rather than the rule, as the effort involved on the part of the virus author is not equal to the added virulence of his creation. That balance would change as any one product became widespread.

None of the above arguments mean that *Symantec's* anti-virus products are predestined for failure - there is no reason why the union should not bear truly golden fruit. However, with this marriage, the company has acquired its own very special problems: whether *Symantec* can live up to users' expectations, and avoid the pitfalls, remains to be seen. There is an ancient Chinese curse: 'May you live in interesting times' - the *Symantec* design team may find these words particularly appropriate.

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

Central Point Merger Completed

In a little-publicised arrangement, *Central Point*, which announced an intention to merge with *Symantec* on 4 April this year, has moved its offices to *Symantec* headquarters.

The name *Central Point* now exists only as the name of its products, and as a subsidiary of *Symantec*, similar to *Fifth Generation* when it was subsumed into *Symantec* last year. The merger, worth approximately US$60 million, took place on 2 June 1994, after the requisite government checks and investigations. Gordon E. Eubanks Jr., the president of *Symantec*, said, 'By merging with *Central Point*, *Symantec* strengthens and increases its resources to build a strong, highly successful enterprise software company.'

Offices worldwide have amalgamated: all *Central Point* offices have now closed, and staff and equipment have moved to *Symantec* buildings.

Central Point Anti-Virus will continue to be marketed for at least the next year, according to Tori Case, product manager for *CPAV*. At the end of this time it will merge with *NAV*, *Symantec's* own product. It has not yet been decided whether the new product will retain one of the two current names, or have a completely new one.

When the new product emerges next year, the objective, said Case, is to make the crossover as easy as possible. There will be a time overlap before the old product is phased out and the new one takes over, and existing customers of both *CPAV* and *NAV* will be offered the opportunity of upgrading to the new product. The exact content of the new product has not yet been decided - the design team, Case explained, is already hard at work on the basics: 'It may even be two different products, aimed at two different markets. *CPAV* will continue to be marketed and supported until that time.'

'Upgrades *will* be honoured to all *Central Point* corporate customers who have site licences,' Case stated with firmness. 'We will take care of all our customers.'

Information on the company and its products can be obtained, in the continental US, on 1 800 441 7234. Outside this area, call +1 503 334 6054 ▮

Pathogen Author Arrested

After a detailed investigation, officers of *New Scotland Yard's Computer Crime Unit* (*CCU*), in collaboration with the *Devon and Cornwall Constabulary Fraud Squad*, have arrested the man whom they believe may be the 'Black Baron'. He is alleged to be responsible for three computer viruses, Smeg.Queeg and Smeg.Pathogen (see *VB*, May 1994, pp.9-11), which recently caused a media panic, and Germ (aka Baron: see *VB*, April 1994, p.4). All three viruses have been reported in the wild. Germ appears to be a relatively simple virus, but both Pathogen and Queeg are encrypted using SMEG, the 'Simulated Metamorphic Encryption Generator'. The level of polymorphism produced by this engine is very high, and most products have some difficulty in detecting the viruses.

The person arrested has been released on bail. Detective Sergeant Simon Janes, of the *CCU*, said, 'There are many enquiries still to be made, and it is likely to be several months before any recommendations are made to the *Crown Prosecution Service* as to further proceedings. At this stage, we are not looking for anyone else, but from an investigative point of view, the case is still open; we do not exclude the possibility of further arrests.'

Janes urges anyone who has suffered attacks from Pathogen, Queeg, or Germ to contact the *Computer Crime Unit* immediately, on +44 (0)71 230 1177 ▮

Virus Prevalence Table - June 1994

Virus	Incidents	(%) Reports
Form	18	29.0%
Parity_Boot	5	8.1%
AntiCMOS	4	6.5%
JackRipper	4	6.5%
Monkey_2	4	6.5%
CMOS4	3	4.8%
Spanish_Telecom	3	4.8%
NoInt	2	3.2%
PS-Dropper	2	3.2%
V-Sign	2	3.2%
Amse	1	1.6%
CMOS1	1	1.6%
Eykad	1	1.6%
Father	1	1.6%
Flip	1	1.6%
Halloween	1	1.6%
Jimi	1	1.6%
Joshi	1	1.6%
Keypress-1216	1	1.6%
Natas	1	1.6%
New_Zealand_2	1	1.6%
Nomenklatura	1	1.6%
Pathogen	1	1.6%
Tequila	1	1.6%
Viresc	1	1.6%
Total	62	100.0%

IBM PC VIRUSES (UPDATE)

The following is a list of updates and amendments to the *Virus Bulletin Table of Known IBM PC Viruses* as of 19 July 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

Agena

CER: A 723 byte virus. Awaiting analysis.

```
Agena           813C 4D5A 7524 8B44 0803 4416 B910 00F7 E103 4414 83D2 0092
```

Australian_Parasite.635

CR: Detected with the Australian_Parasite.615 pattern.

Dark_Avenger

CER: In August 1992, a pattern was published for the Dark_Avenger.Father virus. It soon became obvious that this pattern was much too generic, detecting not only Dark_Avenger-related viruses, but also a large number of viruses sharing certain code fragments. The following pattern can be found in four recent viruses: Dark_Avenger.Shyster (1802 bytes), Rape.1882, Rape.1885 and Rape.2887.

```
DA-related      C31B D172 0429 0606 005E 561E 0E33 FF8E DFC5 069C 002E 8984
```

Demolition.B

CR: Detected with the Demolition pattern.

DIR_II

LR: The J, K and L variants are detected with the DIR_II.A pattern.

Firefly

CN: A 1106-byte encrypted virus which can be detected with a searchstring containing wildcards. When decrypted, the virus contains several text strings, including '[Firefly] By Nikademus', 'Every day is Halloween' and 'Happiness in Slavery'.

```
Firefly         BB?? ??B9 1001 8137 ???? 8177 02?? ??83 C304 E2F2
```

Genc

CR: There are two viruses in this family. One is 502 bytes long, and contains the text 'This virus is Shaware!'; the other is 1000 bytes and contains 'GencVir (C) 1993 by HACKER'.

```
Genc.502        3D00 4B74 0B3D 003D 7406 9D2E FF2E C202 9D2E 8C1E C802 0E1F
Genc.1000       3D00 4B74 069D 2EFF 2EB6 049D 2E89 26BC 042E 8C16 BE04 5053
```

Genesis.501

CR: Detected with the Genesis pattern.

Genvir

CN: There are several viruses which have been created with the GenVir construction tool, or a 'hacked' version of it. Two of these (1376 and Wednesday, 1312 bytes) are overwriting, and detected with the Genvir-over pattern below. The other pattern is a 'generic' Genvir pattern, which will detect the following Genvir-generated viruses: 1440, 1800.A, Cousin, Gomez, Lurch, Morticia, Pugsley, Thing and Uncle.

```
Genvir          B440 41CD 218B C3EB D62E A122 0133 F650 8B5E 10B8 0002 33D2
Genvir-over     B440 2E8B 0E22 01BA 0001 CD21 7307 2E8B 1E2C 01FF E3B8 0157
```

Gidra

CN: A 469-byte virus, containing the text 'I'm GIDRA v1.6 : Life is Good, But Good Life Better Yet.'

```
Gidra           B440 8D94 4F01 B9D5 0190 CD21 7303 E954 FFB8 0042 33C9 33D2
```

Gippo

ER: A family of Italian viruses, most of which are encrypted. The known variants are Gippo.Bumpy (1039 bytes), Gippo.Earthquake (1000 bytes), Gippo.Epidemic (1242 bytes), Gippo.JumpingJack (901 bytes, not encrypted), Gippo.Sunrise (1030 bytes) and Gippo.Stunning (1234 bytes).

```
Bumpy           5053 5152 1E06 0E1F BE2F 00B9 F701 8B04 BA?? ??0B C2F7 D021
Earthquake      5053 511E 068C C88E D88C 0693 0483 3E95 042A 740C B9DB 01BE
Epidemic        5053 511E 060E 1FB9 5A02 ??BE 3200 ??8B 1CBA ???? 0BDA F7D3
JumpingJack     BA10 002B D181 C185 03B4 40CD 2158 8826 9403 B900 218B 16B7
Sunrise         5053 511E 060E 1F8C 06C1 0483 3EC3 042D 740D BF3F 00B9 EB01
Stunning        5053 5152 1E06 0E1F BE30 00BA ???? B958 028B 040B C2F7 D0F8
```

Goga

CEN: A 1660-byte virus. Awaiting analysis.

```
Goga            B440 CD21 EB25 8B1E 8006 33C9 33D2 B802 42CD 21D1 E8D1 EA73
```

Gollum

CN: A variable-size virus, bearing some resemblance to Vienna.

```
Gollum          8DB4 1C00 BF00 01B9 1C00 F3A4 5E56 8DB4 0300 8BFE B919 00AC
```

Intruder.1317 **ER:** Detected with the Intruder pattern.

Ionkin.195 **CN:** Detected with the Ionkin.218 pattern.

IVP **Various:** Several new IVP-generated viruses have appeared recently. As in the case of PS-MPC and VCL-generated viruses, search patterns containing wildcards are not provided for the viruses - these viruses should be detected with a scanner capable of handling all possible viruses generated by the toolkit. Recent IVP-generated viruses include: 351, 540, 644, April, Dread, Mandela, Ozzy, Panic, Stress, Swank, Taselhoff, Tim, Wild_Thing.555, Wild_Thing.557, and Wild_Thing.567

Khizhnjak.377 **CN:** Detected with the Khizhnjak.642 pattern.

Leprosy.I **CN:** Detected with the Leprosy C2 pattern.

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

Necropolis.B **CER:** Detected with the Necropolis (1963) pattern.

November_17th.900.C **ER:** Detected with the November_17th.900.A pattern.

Npox **CER:** Two new variants, Npox.955 and Npox.1015, both detected with the Npox pattern. These viruses are also detected by the ZK-900 pattern, and the ZK-900 virus has been reclassified as Npox.900.

PCBB.833 **CR:** Detected with the PCBB.1129 (Plaice) pattern.

Proto-T **CER:** Two new variants have been found, both of which are detected with the Proto-T.Ritzen.1087 pattern. They are Proto-T.Ritzen.1098 and Proto-T.Ritzen.1112.

PS-MPC **CER, CN:** There are surprisingly few PS-MPC-generated viruses this month, only G2.Dread (CER, 612), Ranger.423 (CN) and Screen_Save (CN, 1207).

Screen+1.919 **CER:** Detected with the Screen+1 (948) pattern.

Stardot.789.D **CEN:** Detected with the Stardot patterns (originally named Sept_18 and 801).

Storm.1219 **CR:** Detected with the Storm pattern.

Tack **CN:** Three new variants have been found, 411, 460 and 477 bytes long. The 411- and 477-byte variants are detected with the same pattern.

```
Tack.411/477        5850 0500 01A3 3E02 C706 4002 FFE0 C606 4202 23B4 408B 1E35
Tack.460            5850 0500 01A3 4702 C706 4902 FFE0 C606 4B02 23B4 408B 1E3E
```

Tajfun **CN:** A simple, 250-byte virus, containing the text 'tajfun1.0'.

```
Tajfun              B440 B9F5 008D 9605 01CD 21B4 4232 C033 D233 C9CD 21B4 40B9
```

Tamanna **CER:** A 1857-byte virus. Awaiting analysis.

```
Tamanna             2680 3E00 005A 7507 263B 0601 0075 E1C3 06E8 D1FF 8B1E 0400
```

Tamsui **ER:** A 1694-byte virus, which contains the text 'Merry Christmas and happy new year! Written from Tamsui Oxford college.'

```
Tamsui              B821 25CD 21B4 2A9C FF1E E205 80FE 0C75 3180 FA17 762C 80FA
```

Tankard.542 **CR:** Very similar to the 556-byte variant reported in February 1993.

```
Tankard.542         80FC FF74 1480 FC3D 7413 3D00 4B74 0E3D 006C 740E 2EFF 2E7C
```

Taurus **CN:** A Polish 358-byte virus which contains the text 'TAURUS (C) Prymityw 0.3'.

```
Taurus              03FE 4789 05B4 40B9 0300 BA5A 0203 D6CD 21B8 0242 33D2 33C9
```

Tolbuhin.992.B **CN:** Detected with the Tolbuhin.1147 (SK-1147) pattern.

Trojector.1561 **CER:** Detected with the Trojector (formerly called Athens) pattern.

Vbasic.D **CEN:** Detected with the Vbasic (5120) pattern.

VCL **CN, PN:** This month brings the following VCL-generated viruses: 355 (PN), Black_Death (CN, 780) and Blue_Moon (CN, 932).

Veronika **CER:** A fairly advanced 1549-byte stealth virus from Russia, containing the string 'Veronika P.'

```
Veronika            5E83 EE0B 0616 FAB8 BBFB CD21 FAFC 179C 5BD0 DF2E D194 FC05
```

Vienna.533 **CN:** A 533-byte variant which will occasionally overwrite files with the old 'reboot' code, instead of infecting them normally.

```
Vienna.533          FC8B F283 C600 90BF 0001 B903 00F3 A48B F2B4 30CD 213C 0075
```

Vienna.608.B **CN:** Detected with the Interceptor, Dr_Q and Vienna-4 patterns.

INSIGHT

Schwartau on Security

Editor of the *Security Insider Report*, author, consultant, crusader against 'virus busting', and well-known conference speaker - Winn Schwartau is a man of many faces, and makes a habit of looking at things from perspectives which would not even occur to most people.

He has adapted a term first coined by Robert Buckminster Fuller to describe his way of looking at the world and what is in it: ephemeralism, which means doing more with less, becomes technological ephemeralism for Schwartau. This requires viewing things from a different perspective to the ordinary, in order to gain a different answer to the ordinary.

Inside Security

His current project, now in its third year, is the monthly newsletter *Security Insider Report*. Schwartau's editorial style is often acidic and biting: he aims to find a different vantage point from which to view issues, and is deliberately as thought-provoking as possible. Showing people an alternative approach is integral to his *modus operandi*.

Part of *SIR's* brief is to combine different aspects of the security world. 'If you say you're into viruses, people say "That's security". It's one aspect,' says Schwartau. 'When I think of security, I think of information security, and put everything under one big umbrella, and see how they interrelate. They're all different aspects of the same thing.'

His literary aspirations do not stop with the *SIR*: he has also written a novel, *Terminal Compromise*, a fictional account of how vulnerable any technical society is to a non-military attack, and a second, non-fiction book, entitled *Information Warfare: Chaos on the Information Superhighway*. His plots feature not bombs and bullets, but the use of information and information systems as weapons and as the targets of those weapons.

Schwartau believes that America will come under threat of a situation similar to that enacted in his novel: 'Unless we do something about it, it's inevitable that we will encounter information assaults. How big, I cannot predict - maybe not as big as in the novel, but it will happen.' This inevitability, he feels, lies in the fact that crimes such as industrial espionage go basically unpunished, as the law has yet to catch up with the crucial value of information.

One of Schwartau's principal concerns is that we are adding connectivity too far, too fast: 'We are building a national information infrastructure: I could compare it to your *European Space Initiative*. We are attempting to build tomorrow based on technology instead of on a well-thought-out policy. There's not only one or two issues, there's hundreds, and they have to come under a single domain. There is also a problem with the fact that the US, like most Western European countries, belongs to the "Information Age": we came into this era on the day our reliance on the econo-technical structure - the wires, the networks, the communications, now the computers - exceeded our ability to live without them. A certain vulnerability has grown, and will continue to grow, because we're building all these systems without proper consideration of the risks.'

Penetrating Observation

To Schwartau, viruses are just a subset of malicious software, which would require extraordinary effort to make them successful against a determined defence. However, he maintains that security in most companies is inherently flawed, allowing viruses to enter. Schwartau's company offers a service called Penetration Testing, which offers analysis of security within a company. From conducting such tests, he knows how easy it is to infect a system.

> *"fortunately, most virus authors don't want to wait too long; they want instant gratification"*

Schwartau did such analysis for a company with a defence contract, which believed that their system was immune to penetration. He and his colleagues planned a remote system invasion, but decided to plant the viruses physically.

'We walked into the contractor's building with a bottle of water to refill the water cooler. We went at lunchtime, so several people would be out, put the water in its place, then loaded some Trojanised software onto office PCs. When people came back and touched their machines, their machines were no longer functional: we had planted an infinite-loop TSR which did no damage, but got the point across very effectively. Physically, it's easy to infect a company.'

Virus Busting

Schwartau disagrees with traditional 'virus-busting', the virus-specific approach currently used by most companies. He sees this method as outdated, and little more than a distribution system in need of constant replenishment; relying for help on 'yesterday's answer'.

'Manufacturers do the best they can,' he asserted, 'but I think that that technique alone is lacking: I advocate security modelling as a technique to keep viruses out of the system and to contain them actively. This approach is not designed to replace scanners, but to add to their effectiveness. Whenever new data comes into or leaves a company, it would still be checked for viruses.'

Security as a Model

In order to move away from the reactive cycle of 'virus busting', Schwartau believes in 'security modelling', which he defines as 'the isolation of system resources through access control, an authentication system so that you always know who is doing what, so that if a virus gets on to a system, you have a history of its origins, and an audit trail.'

Schwartau's justification for this type of security policy is that very few people actually need control of all the executable files and various other types of system files in anything other than an execute mode. He believes it would be easy to enforce such a policy as a protective device.

The core of this hypothetical system is the Reference Monitor, which traps all operations on the system, and makes one of two responses to a request: Go (the event may proceed as requested) or No-Go (the process is stopped).

In such a model, every user is identified and authenticated each time he accesses the system; upon admission, there are controls on what he can and cannot do; when he leaves, all traces of his activity will be erased from RAM. Audit trails are kept at all times, and confidentiality is prioritised - encryption prevents unauthorised users from reading files.

Schwartau admits that this model is not bullet-proof, but claims it would limit damage and disruption. 'If something did get through, it would be isolated on one machine, and there would be a history of exactly when and where it occurred. With these mechanisms in place,' he explained, 'chances of major damage are minimised.'

On People and Propagation

One of the issues at the forefront of people's consciousness in the past few months has been the writing and releasing of viruses; specifically, the announcement of Mark Ludwig's latest virus-writing contest, and the release of his CD-ROM containing many thousands of live viruses.

In Schwartau's opinion, this event is not a major issue: 'I don't have the sense of moral outrage from the virus issue I see voiced by others: I save that for serial murderers, abused kids and undeserved poverty,' he said. 'Mark Ludwig is just some guy, writing about a virus-writing contest. There were thirty entries last year. Ignore it!'

Isn't the CD-ROM taking the freedom of speech guaranteed by the First Amendment too far? Schwartau thinks not: 'The First Amendment is not entirely understood by non-Americans. They don't realise how strongly we feel about our right to say or write what we please. Few people, no matter how they feel about viruses or Mark Ludwig, would disagree with the fact that we Americans have the right to write a software program that does anything whatsoever. We do *not* have the right to do something harmful with that virus. However, we do not know how to achieve a legal way of expressing that. How do we formulate something which gets the bad guy, but leaves the good guy alone?'

Winn Schwartau: 'I don't have the sense of moral outrage from the virus issue I see voiced by others: I save that for serial murderers, abused kids and undeserved poverty.'

Looking to the Future

In Schwartau's opinion, viruses will probably go the way of many other types of 'high-tech crime': more incidents, with less impact. As people become more aware, there will be more involvement in containment, so despite greater total effect, the individual effect will be less strongly felt.

He also warns against the possibility of delayed-reaction payloads: 'Let's say three years ago, one of these virus-writing kids thinks, "Wow, maybe I could get shareware out there that doesn't do anything bad for five years." So, he writes some good software, and it gets distributed. At some point, it becomes a virus… Fortunately, most virus authors don't want to wait too long; they want instant gratification.'

Schwartau's own belief is that, if someone wanted to wreak maximum havoc, it would be easier, albeit more expensive, to buy a software company, and do the damage through that: 'Distribution would cost you maybe 30 to 40 million dollars, but the effect would be dramatic.'

He sees malicious damage, through viruses and other means, becoming ever more commonplace in the future. Security modelling, in his opinion, is the only sensible way forward. Minimising the chances of a virus getting into the system, containment in the event of its success, and mechanisms to provide a history of exactly how an offending item managed to subvert stringent controls - these are what Winn Schwartau's concept claims to offer.

He readily admits that anti-virus vendors have no time for such a departure from tradition: 'This is because instead of selling thousands of their products, they would only sell a handful!' Anti-virus software, in his opinion, comes complete with built-in obsolescence: can the view of millions of users be so misguided? Time will tell.

Winn Schwartau and *SIR* may be contacted on
Tel. +1 813 393 6600 or Email p00506@psilink.com

FEATURE

CARO: A Personal View

Fridrik Skulason

The past five years have seen many attempts at forming anti-virus organisations. I have watched them come and go, and seen many replaced by other groups with slightly different goals. Often, however, 'core' participants remain the same - the number of people in this field is rather limited.

One organisation was *AMC* (*Anti-virus Method Congress*), a short-lived attempt to unite developers and users which fell apart the day it was formed. Then came *VSI* (*Virus Security Institute*), an organisation of researchers and developers which attempted (and failed) to hold a virus conference - and is now reduced to an almost inactive mailing list.

AVPD (*Anti-Virus Product Developers*) was, as its name indicates, an organisation of companies within the industry. It may still exist - I personally lost interest in the group some time ago. Others include *NCSA* (*National Computer Security Association*), *ICSA* (*International Computer Security Association*), *CVIA* (*Computer Virus Industry Association*) and *EICAR* (*European Institute of Computer Anti-virus Research*). Some still function: not all are limited to computer viruses; some deal with security in general.

There is an organisation of *Macintosh* virus experts, which seems to be trying to keep its very existence, or at least its members' names, secret. Finally, there is *CARO* (*Computer Anti-virus Research Organisation*). These last two bodies are different from those mentioned above, actually doing things to benefit their members, and, indirectly, the whole user community. I will not attempt to describe the *Macintosh* organisation, but as a founding member of *CARO*, I should be qualified to explain what *CARO* is - and is not.

CARO: The Beer-Drinking Club

CARO members have always made it clear that the group is not an industry association. It might best be defined as an informal organisation of people (*CAROts*) who get together every now and then, drink beer, eat pistachios, try Chinese restaurants all over the world (ever wondered why some of us are slightly overweight?), and chat about such subjects as computer viruses and uses of leftover military hardware. Between beers, *CAROts* exchange virus information, or even live samples.

CAROts live in every corner of the globe, and can rarely sit down together, so we frequently correspond by Email. *CARO* is not officially registered anywhere and has no membership fees, no formal charter of operation, minimal overheads: exactly how it should be. The most formal organisations in this area have also been the most short-lived, and the worst waste of time for all involved.

An Organisation of Individuals

CARO is an organisation, not of companies or company representatives, but of anti-virus authors and researchers, some of whom work for companies producing anti-virus soft- and hardware. In some cases this distinction does not matter. Some members run (or used to run) a single-man company; others work for companies with huge legal departments (it would be more difficult for such people to join as official company representatives than as individuals).

If a *CARO* member switched companies, he would almost certainly remain a member, the company he left having no right to appoint a 'replacement'. In fact, *CARO* participation is not always actively supported by companies for whom members work - marketing departments do not always seem to like the idea of their technical people meeting with the competition over a glass of beer (many glasses, in fact…). Other *CAROts* do not work for anti-virus companies at all; for example, those at universities.

CARO Activities

Ignoring the beer-drinking and other activities which have nothing to do with viruses, *CARO* activity falls into one of five categories: virus-naming, virus descriptions, the *CARO* WildList, exchange of viruses (or virus information), and the *CARO* mailing list. Many of these benefit, at least indirectly, the user community.

Within *CARO*, a small naming committee (Alan Solomon, Vesselin Bontchev and myself) is responsible for selecting 'official' names for new viruses. *CARO* has no power to force anti-virus companies to adopt these names, but we do our best to encourage it: this would help to reduce the confusion caused by the use of multiple names for one virus.

There has also been work on a database of virus descriptions, called *CaroBase*. This is intended to provide a more accurate alternative to *VSUM*, but has not yet reached distribution stage. If and when it does, the benefits will be obvious - there is a need for an extensive, accurate virus information database.

The WildList is (just as the name suggests) a list of viruses 'in the wild', kept up to date by Joe Wells, who works for *Symantec*. It collates reliable reports from all over the world on virus frequency and incidents. It cannot be 100% accurate, but is the best list of its kind currently available.

When *CAROts* meet, they may exchange recently-received viruses and various bits of virus-related information. Meeting in person often involves setting up a small LAN: one *CAROt* brings a portable *NetWare* server and the rest bring laptops, adapters, T-pieces and short cables. Everyone participating uploads his material, and downloads the rest.

There are several mailing lists for use by *CARO* members, for technical purposes. These are closed to non-members, but one (vquery@rz.uni-karlsruhe.de) enables interested parties outside *CARO* to send queries to members.

The *CARO* Collection

One often hears about this: sometimes a computer magazine will request access to it, and there have been cases of someone claiming to have obtained it. However, the truth is that there is no such thing as a *CARO* virus collection. Most members maintain their own collections, and although they may be similar, they are certainly not identical. Of course, some are bigger or better organised than others - the best collection is probably that of Vesselin Bontchev in Hamburg, but even this cannot be called 'The *CARO* Collection'.

Joining CARO

When *CARO* was formed on 10 December 1990, there were fewer than ten members, but today there are nearly 30 *CAROts*. Joining *CARO* is not a simple matter of signing a form and paying a membership fee. Some new members are invited; others apply and pass the voting process. Existing *CAROts* vote on candidates, and each member has the right to veto any application. Even if nobody rejects a candidate, a certain percentage of members must actually vote for him, instead of abstaining.

Does this sound harsh? Maybe, but keep in mind that *CARO* is not an industry association which does not care who the members are, as long as they pay their annual membership fee. This is a group of individuals who trust each other, and who must be able to do so: we regularly exchange sensitive information which we want to prevent falling into the wrong hands. Although this does not imply that members have to like each other, it is usually the case that we do.

An application may be rejected for several reasons, but some are more common then others. For example, the 'Who's that?' problem: there have been a few cases where applications were received from people few *CAROts* knew personally or with whom they had corresponded. Such applications generally failed because too many *CAROts* abstained.

Any application from known virus authors, anyone involved in unrestricted virus distribution, encouraging virus writing or behaviour considered unethical by *CARO* members will be rejected without consideration. In addition, we expect a certain level of viral knowledge, as well as several years' experience in the field. Applications from those interested only in collecting viruses are rejected immediately.

There have been occasional accusations that this makes us an 'elitist' club… there may be a grain of truth in that, but this system has kept *CARO* working for several years, and enabled us to get some useful work done, as well as having great fun between glasses of beer. [*The next CARO meeting is planned for the VB Conference in Jersey. The bar has already been informed. Ed.*]

VIRUS ANALYSIS 1

AntiCMOS - Brain Damage

Derek Karpinski
Andersen Consulting

The research community has been aware of the existence of AntiCMOS as a 'laboratory specimen' for some time: it is now, however, in the wild, having recently arrived in the UK from Italy. As a result of safe practice, it was detected before it infected anything, and then drawn to my attention.

Plus ça Change…

AntiCMOS is an exceptionally primitive boot sector virus, and infects both hard and floppy disks. Unusually for this type of virus, the original boot sector of infected diskettes is not stored anywhere, though a substantial amount of virus code appears dedicated to finding a place for it.

Overall impressions are that this is an extremely poor attempt at virus-writing. The author appears to have given up halfway through - hardly surprising, as he seems incapable of producing simple code. Diskettes infected by the virus will no longer be bootable, although they are still able to infect a hard drive.

The virus has virtually no error checking, makes no attempt to check if a disk is already infected, is obvious in operation, and is easy to detect and remove. However, it does replicate, it is in the wild and it has a particularly annoying payload.

On Booting

The virus creates a stack for its own use and stores the Int 13h disk handler interrupt vector in the viral image in memory. Two Kbytes at the top of available low memory are reserved by the virus so that its code will not be overwritten: the resultant memory loss is easily detectable. The virus then copies itself into this protected area of memory, and relocates to continue execution from there. Next, the disk controller is reset, and the virus examines a data area within itself to determine if the machine was booted from the floppy or the hard drive.

If the machine was booted from the hard drive, the virus will find the current active partition and copy its boot sector into memory. The data area in memory which determines the boot drive type is set to the value for a floppy drive, and a replacement Int 13h handler is installed. Control is then passed to the boot sector for the current active partition, and booting continues normally.

If the machine was booted from an infected floppy, the virus attempts to infect the Master Boot Sector (MBS) of the first hard drive. A single subroutine is used to infect both floppy and hard drives.

After infecting the hard drive, the virus begins a series of calculations based on the number of entries in the root directory, the number of sectors in the File Allocation Table and the number of sectors per track. This is presumably intended to identify the location of the original boot sector, had it been stored. At several points during this process, the virus can cause the system to hang.

A replacement Int 13h handler is then installed, and the virus passes control to the memory location into which a boot sector is loaded: this area still contains the virus code. Thus, the system will hang when booting from a floppy, even if this has been avoided so far.

Infection Routine

The infection process is identical for both floppy and hard drives. The contents of Track 0, Sector 1, Head 0 (the boot sector of a floppy, or MBS of a hard disk) are read into a buffer. The virus then overwrites the initial jump instruction in this buffer with its own value and copies the remaining virus code into the buffer. However, it does not overwrite the boot sector data or the partition table in this buffer (if present). Next, it writes its code to Track 0, Sector 1, Head 0 of the floppy or hard drive, but makes no attempt to retain the original boot sector of a floppy disk.

The replacement Int 13h disk handler represents more bungled coding. It checks to see if the requested access is to the hard or floppy drive: if the former, no further action is taken. If for a floppy, the handler attempts to determine if it is a read or write request. Regardless of the type of request made, the virus then takes the high nibble of the least significant timer count maintained by the BIOS, and subtracts the value of the byte at offset three of the MBS (remember - this virus will only be active after booting from the hard drive). If the result is less than two, the trigger routine is called (see below).

Thus, triggering cannot be predicted, although it occurred with monotonous regularity during my experimentation. If the result is two or more, the diskette is infected. No check is made to see if the disk is already infected, no attempt is made to hide the operation by avoiding infection if the drive is already running, and no error checks are made. One side effect is a painful slowdown when accessing floppy disks.

Trigger Effects

The payload effectively destroys the data stored in the CMOS memory, which typically holds information on system configuration (including base and extended memory size), the type of disks installed, the primary display and the maths coprocessor.

Thus, when the PC next starts, it will go straight to BASIC in ROM BIOS (for genuine *IBM-PCs* with ROM BASIC): in essence, it will forget that it has a hard drive. The payload is annoying and highly visible; it does not physically destroy data, but may cause people to think their data has been lost.

If the infected machine is an *IBM-PC* with microchannel architecture, recovery is easily done by booting from a reference diskette for that machine, and using the automatic configuration feature. For other machines, the documentation supplied with the machine should be consulted. Of course, this is often easier said than done.

Removal

The virus does not store the original boot sector, making removal from the hard drives of machines formatted pre-*DOS 3.31* problematic. For machines formatted with *DOS 3.31* or later, boot from a write-protected system floppy and use the FDISK /MBR command to restore the original MBS. For pre-*DOS 3.31* machines, copy the MBS with an appropriate utility, then boot from a *DOS 3.31* or later disk and attempt FDISK /MBR: not a guaranteed fix, but if it does not work, there is still the backed-up MBS.

Copying a clean MBS table from an identically-configured PC, and using a disk editor to replace the partition boot table correctly, is another option. Removal from a floppy entails simply using the SYS command under clean conditions.

In Conclusion

I had to check and recheck my work very carefully during this disassembly, as I found it difficult to believe that even a 'tyro' virus writer could produce something quite so poor. It is almost easier to believe that it was produced by ten thousand monkeys playing with a keyboard. It is disturbing to think that this virus has 'escaped' into the wild - it is so obvious, and so easily detectable, that anyone with an ounce of sense and/or an average scanner could find it. AntiCMOS is almost more of a Trojan than a viable virus, although it can (just) replicate. Seek and destroy.

AntiCMOS

Aliases:	None known.
Type:	Memory-resident boot sector virus.
Infection:	Master Boot Sector of hard drive, boot sector of floppy disk.
Self-recognition in Memory:	None.
Self-recognition on Disk:	
Hex Pattern:	`8826 0300 3D02 0073 03E8 CC00 E8E8 0058 1F2E FF2E 0700 33C0`
Intercepts:	Int 13h for infection.
Trigger:	Overwrites CMOS RAM data area.
Removal:	Under clean system conditions, use the FDISK /MBR command. For further details, see text.

VIRUS ANALYSIS 2

A Pile of Junk?

Mike Lambert
Rochester Telephone Corporation

Well, Junkie came and went, and I guess I must have missed the boat. I took a look at the virus when I was first sent a copy and, getting bored, went on to another. Surely this was no big deal: no stealth, no special encryption (does anyone who is not 'hardcoding' polymorphic detection routines even care any more?), no trigger routine… the sort of virus which I receive by the gross every month.

All of a sudden there are warnings of Junkie on the public nets! To listen to Cyberspace, one would think Junkie was 'the mother of all viruses'! Somewhat chastened by what I was being told about Junkie, I took a look at it a second time to see if I had missed the point. I had not: Junkie is by no means a new wonder-virus. All that noise and it doesn't even infect a 360K floppy! This thing can be described in a single breath.

A Pressing Problem

The reasons for the virus' infamy are all too familiar. The story is that a *Reflex Inc.* representative found it in Ann Arbor, Michigan using *DiskNet*. It would appear that the virus was taken very seriously (by whom, I'm not too sure!). The press picked up the *Reflex* press release and ran with it, and the unknowing finally read about it in the newspapers and on the newswires. The rest is history.

Some may view some of the virus' techniques as new or innovative, but I see nothing revolutionary about them. If you missed this virus, it is nothing to worry about unless you are using a substandard product, rely wholly on scanners, or cannot restore a Master Boot Sector (MBS) and replace infected COM files. There is no attempt to hide the virus, and no destructive trigger routine.

In short, Junkie is a parasitic COM infector, dependent on the boot mechanism to establish residency to infect Boot Sectors and proliferate via COM files. The virus contains no stealth routines, is easy to spot, uses simple encryption, has no trigger, is incapable of advanced penetration techniques, and targets the TSR components of both *CPAV* (version 2.0) and *MSAV* (as shipped with *MS-DOS 6.2*).

Virus Operation

The infection routine used by the virus is simple, limping along, years later, in the footsteps of Tequila (the EXE multi-partite of days of old which 'introduced' multi-partite viruses). On COM execution, Junkie only drops the virus on drive 80h (see more below) and transfers control to the host. It does not go resident on COM execution (I use the term 'integrated' for those multi-partites which are equally infectious when loaded from files or a boot sector) so even elementary software write-protection is an absolute deterrent. Initial infection consists of dropping the virus loader in the real MBS code (16 words) and the virus body in sectors 4 and 5 (Head 0, Cylinder 0). The current Interrupt 13h vector in the interrupt vector table is used to make the call to carry out this write, which is why any software write-protection is effective. There is no check made as to whether boot sector infection has been successful.

"chastened by what I was being told about Junkie, I took a look at it a second time to see if I had missed the point. I had not…"

At system startup, the virus code in the MBS loads the two additional sectors of virus code and transfers control to the virus decryptor. Once this operation is complete, the virus hooks Interrupt 13h for later floppy infections, and Interrupt 1Ch (the timer) in order to hook Interrupt 21h after DOS has loaded. The timer interrupt is flag-driven, and never unhooked from the system.

This flag allows one to enter the virus' timer tick interrupt handler easily, located at 9F40:01EA on a 640K system (on such a system, the memory-resident copy of the virus resides at 9F40:0000, and uses the 'standard' boot sector virus memory-stealing technique). When resident, the virus occupies 3K of memory. If CHKDSK is run on an infected machine, DOS returns 652,288 of base memory, infecting CHKDSK.COM in the process.

Other indicators of the virus being active in memory are that IO.SYS will use 9F40:0091 for disk services, and the Int 21h vector will point to near the top of memory, 9F40:0237.

Boot Sector Operation and Removal

The virus does not store a copy of the uninfected MBS, but opts to restore the 16 overwritten words of the loader routine with the original code just before returning control to the MBS. The original boot sector from both the fixed disk and any infected diskettes is not replaced on access, so there is no attempt at stealth. As a result, it is impossible to restore the original MBS by relocating a copy stored by the virus, or by allowing the virus' own stealth routine to restore the MBS itself.

In order to recover from the virus, one must either restore the real MBS from a backup copy or rewrite the MBS loader using the DOS command FDISK /MBR. Many anti-virus products also include a utility to restore such damage.

Memory-resident Operation

Interrupt 13h processing consists of trapping A: boot sector reads in order to infect 720K, 1.2MB, and 1.4MB (but not 360K) floppies. If an uninfected system is subsequently booted from such a diskette, the fixed disk is infected.

On 720K and 1.2MB diskettes (type F9h), the extra two sectors of the virus code are stored in sectors 8 and 9 of the last track (Head 1, Cylinder 79). This leaves a little extra space on 1.2MB floppies. On 1.44MB floppies, the last two sectors are used (Head 1, Cylinder 79, Sectors 17 and 18). Original boot sectors are not saved, and infected sectors are not protected and can be overwritten.

Interrupt 21h processing consists of trapping Load and Execute, File Open, and Extended Open. During infection, the *CPAV* and *MSAV* TSRs are disabled so the virus can infect files (even this is nothing new - many virus writers know how to do this).

The virus, encrypted with a simple XOR, adds between 1027 and 1041 bytes to COM files over 4K long. COM infection continues as files are opened or executed. The only validation of a COM host is by file extension, so EXE files renamed to COM files (within required size) can host the virus dropper but are destroyed during infection. No 'Are you there?' call is necessary, as the virus only becomes memory-resident when loaded from the boot sector.

A Fast Infector

An important note, which is applicable to a number of different multi-partite viruses, is that it is sometimes necessary to use the SYS command in order to disinfect an infected disk. The reason for this is simple.

It is possible that the operating system installed on the infected machine uses system files with a COM extension (for example, IBM's IBMBIO.COM v3.30). Although these files are suitable candidates for infection, in the normal course of viral operation they will not be infected, because they are 'opened' before DOS has set up its own Int 21h vector. This is a characteristic of the technique used to set the Interrupt 21h vector and holds true for all viruses using the timer tick in order to hook Int 21h.

However, when a disk is scanned with an anti-virus product which fails to detect the virus in memory, every eligible COM file on the disk will be infected. This 'infect on open' strategy (making the virus a 'fast infector'), was one of the points highlighted for special concern in much of the coverage of Junkie. Once again, this attribute is nothing new: the elderly 4K virus (Frodo), does a much more effective job of this, infecting COM and EXE files on Close.

This makes it vital that disk scanning only takes place after a scan of memory using the search string provided, or after booting the system from a clean, write-protected system disk. A good rule of thumb is never to rely on a scanner to check memory - whenever possible, use a cold boot.

Conclusions

The principal lesson to be learnt from this virus is that one should treat popular press virus alerts with a large pinch of salt: Junkie is a long way from being the wonder virus which the press would have had us believe.

If this virus has a purpose, it may be to illustrate that the trailing edge is very rusty. If you can 'catch' this virus, you have no protection at all. If you cannot find this virus, it is time to change your anti-virus software vendor. If you cannot recover from this virus, take action before you face something which is a real threat.

Junkie

Aliases: None known.

Type: Multipartite.

Infection: COM files, MBS of the fixed disk, and the boot sector of 720K, 1.2MB, and 1.44MB diskettes.

Self-recognition in Files:

The length of the file in paragraphs is checked.

Self-recognition on Disk:

The first jump of the MBS is checked, as well as the first word of the loader.

Self-recognition in Memory:

None necessary. The virus does not become memory-resident when an infected file is executed.

Hex Pattern: Due to the short length of these patterns, they should be used with care. Junkie-infected files:

```
BE?? ??B9 F401 2681
34?? ??46 46E2 F7??
```

Junkie-infected MBS:

```
FB8E C7B8 0202 BB00
7EB9 0400 BA80 0056
```

Intercepts: Int 13h for diskette infection, Int 1Ch for hooking Int 21h after DOS has loaded, and Int 21h for file infection.

Trigger: Disables the *Central Point Anti-Virus* and *Microsoft Anti-Virus* TSRs.

Removal: Under clean system conditions, identify and replace infected files. Note the possibility of the system files IBMDOS.COM and IBMIO.COM becoming infected. Use the DOS command FDISK /MBR to remove the virus from the MBS of the fixed disk.

VIRUS ANALYSIS 3

Pure Thoughts…

Eugene Kaspersky

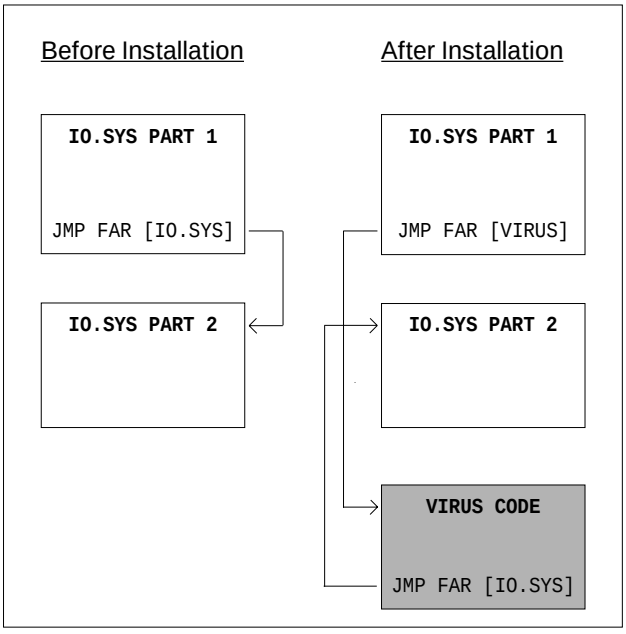
Every day, new viruses are sent to me. Some of them are simple, others are run of the mill, but a few are, from a purely technical point of view, excellent. One good example of a well-written virus is Pure, which crams full stealth and fast infection code into a mere 441 bytes! This virus is a very good example of the experience and ability of an extremely small minority of virus writers: sometimes, the anti-virus researchers have very good adversaries indeed.

Installation

When an infected file is executed, control passes to the virus code, which uses an unusual routine to load Pure into memory. First, a DOS Flush Buffers call (Int 21h, AH=0Dh) is made: this avoids conflicts with certain disk-caching utilities such as *SmartDrive*.

Next, the virus allocates a block of High Memory (from the HMA), using the Allocate HMA call, Int 2Fh, AX=4A02h, before resetting any loaded disk cache software by issuing a Flush Buffers command (Int 2Fh, AX=4A10h).

One point to note is that if DOS is loaded low, the virus will not install itself, and is therefore incapable of spreading. Moreover, the virus disinfects infected files when they are executed under these conditions: this feature can be used to recover infected files if backups are unavailable. Thus, if Pure is executed on a PC XT, it will be incapable of installing itself into memory, and will automatically remove itself from any infected files.



If the HMA block is successfully allocated, the virus copies its code into the HMA, and begins the process of hooking Int 13h. This is carried out by using a method of interrupt stripping similar to that used by Yankee_Doodle: the virus sets up its own Int 01h handler, and issues a call to the Int 13h Verify function (Int 13h, AH=04h) with the trace flag set. After every instruction of the Int 13h handler, control returns to the virus code.

The virus does not examine the address from where the traced code returned (the 'standard' way to trace the Int 13h vector), but checks the value of the next opcode to be executed. If this is FF2Eh (a JMP FAR instruction), and the jump passes control to the HMA, the virus stores the original jump address and substitutes a jump instruction to its own Int 13h handler: this is shown in the box at left. Note that both IO.SYS Part 2 and the virus code is in the HMA.

This method requires further explanation in order to understand fully why it works. On loading high, the DOS programs IO.SYS and MSDOS.SYS are divided into two parts. The first part of each program is loaded into conventional memory, and the second block is placed in high memory. During the tracing process, Pure detects the jump where the first part of IO.SYS passes control to the second.

As a result of the virus' intervention, the conventional memory part of IO.SYS passes control not to its own HMA part, but to the virus' Int 13h handler. This handler then checks the function number of the Int 13h routine called, carries out whatever operations it deems necessary, and passes control to the HMA portion of IO.SYS. Thus, the virus 'wedges' itself between two DOS components, hooking Int 13h without altering any vectors.

Once this tracing routine is complete, the virus disinfects the host file using standard DOS calls (Open, Read, Write, Close and Execute) and the undocumented System File Table. After execution of the host file, the virus searches for fifteen files which fit the mask *.E* and infects them. This infection routine consists merely of opening and reading the files, as the TSR portion of the virus, which is already loaded, intercepts the Int 13h Read calls and completes the infection process.

Infection and Stealth

File infection is accomplished through the interception of Int 13h, which traps three functions: Read (AH=02h), Write (AH=03h), and Verify (AH=04h). The start of each intercepted sector to which the call relates is checked for the EXE file identifier MZ. If this identifier is present, the length of the EXE module is checked by testing the word at offset 04h in the EXE header: if the length is less than 61K, and the virus is not in the New Executable format, the file header is examined for free space. Should all bytes from

offset 0047h to 0200h in the header be zero, the virus will copy itself into that area, and overwrite the beginning of the EXE header with a JMP instruction. Finally, the virus writes the infected sector back to the disk.

The result of the operation described above is to convert the internal format of an EXE file to that of a COM file. This does not prevent normal execution of such files: the virus disinfects the host file if it is executed, and the stealth routine substitutes the original first sector of infected files if the virus is memory-resident.

Hide on Seek

The stealth process is relatively simple. The virus compares the contents of the sector to be read with the virus code, and, if they are the same, the virus overwrites its own code in the data buffer and replaces the MZ identifier. By using this trick, the virus hides itself at a very low level: if an integrity checker is run on an infected machine with the virus operational, no changes will be seen unless the anti-virus software uses direct calls to the BIOS.

The way in which the stealth routine functions also prohibits execution of the virus code, if the virus is already memory-resident. In such cases, when DOS loads an infected file, the virus will disinfect the infected sector 'on the fly'. The virus does not need to use an 'Are you there?' call, as, if the virus is already active in memory, control is never passed to the installation routine.

One final advantage of using the Int 13h hook instead of Int 21h is that the virus does not need to check, save or restore an infected file's time and date stamp or attributes.

The virus does not hook the Fatal Error Handler, Int 24h, during infection, resulting in the familiar 'Write protect error writing drive B Abort, Retry, Fail?' message when a write-protected disk is encountered. An interesting point is that the virus *does* intercept the Int 24h handler during its own installation routine.

Infected files do not become longer after infection, and the size of conventional memory does not decrease once the virus is operational. This, combined with the stealth routine, makes the virus difficult to detect, and a clean boot or an anti-virus product is required to disable the virus in memory.

As already stated, Pure is a fast infector, infecting suitable files on read, write or verify of their file header. Thus, if an EXE file is opened, executed or modified on an infected machine, it will be infected. During copying, both source and destination EXE files will be infected.

Source of Assistance?

Virus writers distribute their creations in many different ways. In some cases, new viruses are uploaded to BBSs under the guise of a new piece of shareware. Sometimes a virus is sent to Virus Exchange BBSs complete with source code, and sometimes it is sent directly to researchers.

In this case, the virus was received complete with its source ASM file, documentation on 'good' features of the virus, instructions on how to disinfect the computer, and the following short description:

```
Virus Name:         PURE
Aliases:
V Status:           New, Research Viron
Discovery:          February, 1994
Symptoms:           None - Pure Stealth
Origin:             USA
Eff Length:         441 Bytes
Type Code:          OReE - Extended HMA Memory
                    Resident Overwriting .EXE
                    Infector
Detection Method:   None
```

Pure is not the first virus to appear with its complete source code, and is unlikely to be the last. Releasing it in this manner does not mean that the virus will spread in the wild, but it does make it more likely that many related viruses will appear, bringing trouble with them. How many Pure variants will there be by next year? We can only wait and see.

Pure

Aliases: ExeHeader.

Type: Memory-resident, parasitic and stealth EXE file infector.

Infection: EXE files only.

Self-recognition in Files:
The virus compares contents of disk sectors with virus' body.

Self-recognition in Memory:
None necessary, due to the virus' own stealth function.

Hex Pattern: Currently three known variants.
Pure.a (440 bytes long):

```
B40D 0D21 33FF 8EDF B702 4FB8
024A CD2F BB06 0047 750A 833E
```

Pure.b (441 bytes long):

```
B40D CD21 33FF 8EDF B702 4FB8
024A CD2F B810 4ABB 0100 CD2F
```

Pure.c (441 bytes long):

```
BA45 59B8 01FA CD16 B40D CD21
33FF 8EDF B702 4FB8 024A CD2F
```

Intercepts: Int 13h for infection and stealth, Int 01h for tunnelling Int 13h.

Trigger: None.

Removal: Under clean system conditions identify and replace infected files. If backups are unavailable, execute infected files on a machine where DOS is loaded in conventional memory.

FEATURE

Epidemic Virus Control

Jonathan D. Lettvin
Lotus Development Corporation

Lotus PC virus control focuses on 'epidemics'. In our experience, it is not the continual stream of new viruses, but epidemics of old, widespread viruses which cause the most damage and lead to the most 'down-time' for communities of PC users. Although it is important that our anti-virus countermeasures keep up with new viruses (which is why we use anti-virus products from many vendors), we focus our internal development strictly on detecting and stopping the viruses which we most frequently encounter.

Lotus defines an 'epidemic' virus in a number of different ways. The most important of these are:

- any virus seen more than once in any group
- any virus seen with a destructive trigger
- any virus from the *Virus Bulletin* prevalence chart

We find some viruses considered 'extinct' by the anti-virus industry to be alive and well. Since new viruses are rare for us, any new virus we encounter at any time, epidemic or not, goes onto our epidemic virus list.

Without argument, Form is the single most widely distributed virus in the industry. Over 80% of all virus support calls we take are from individuals who booted from a Form-infected diskette. The second place in our epidemic list is shared by AntiCMOS and Ibex. Both viruses are capable of major disruptions on PCs running *Windows NT*, as well as on DOS PCs. Other viruses are encountered on a regular basis, including Michelangelo, Ripper, and AntiEXE. All these viruses have, by our definition, epidemic status.

Behaviour Causes Epidemics

Epidemics are caused by human group behaviour, not individual viruses. Of all the problems we try to solve, the most intractable but interesting one is changing group knowledge and habits. Competent anti-virus professionals know precisely what users must do to remove these viruses. They also know what they must *not* do if they want to avoid these viruses - avoid booting from unscanned diskettes. Virus experts know this instinctively; however, it is difficult to get a group of 20, 200, or 2000 regular employees to understand this adequately without threatening firm disciplinary action. This is the hard part of their job.

To address this problem, *Lotus* has developed an epidemic virus disabler/remover program called FIXALL. Although FIXALL cannot prevent the introduction of epidemic viruses onto a PC, it can be used to detect the problem early, remove the virus, and instruct the user on effective actions to take. FIXALL is required as the first line in every DOS AUTOEXEC.BAT, and is completely transparent unless an epidemic virus has been introduced onto the users' PC. After disabling and removing the virus, FIXALL must inform the user of the situation.

Informing users of a virus attack can be problematic. Even when presenting relatively simplistic virus information to a user, we see them panic, ignore it, revert to childhood, or become quasi-experts. At *Lotus* we have found the following formula effective when alerting users. FIXALL tells users which virus was disabled and/or removed, where the virus was found, how their machine was infected, how to eliminate the virus from diskettes, and how to avoid reinfection. Finally, the user is asked to call the *Lotus* virus hot-line to report their name, telephone, location, virus identity, and what version of FIXALL the user has. Calls to the hot-line are satisfyingly brief and confident-sounding.

Criteria for Virus Removal

Due to in-house experience with viruses, *Lotus* has developed criteria for accepting virus removal software. Currently, no product on the market fully meets these criteria.

Lotus requires anti-virus product documentation to prove that the following actions are performed. Any deviation from these actions must be explained and given adequate compensatory action.

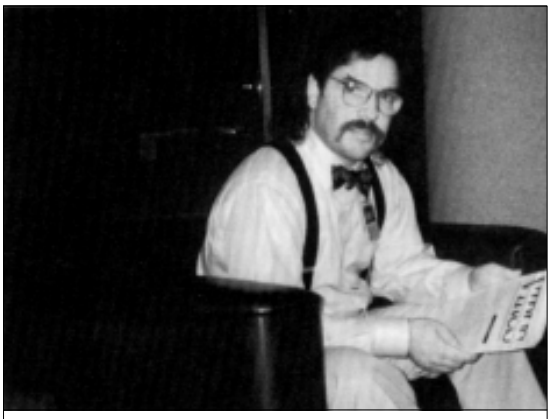
Incomplete removal of any common virus for which complete removal is easy, is unacceptable. Absence of a removal method for an uncommon or new virus is acceptable if the vendor is willing to provide a complete removal on short notice (days, not weeks).

New removal methods must maintain, as much as possible, the file names and calling conventions of their predecessors. This requirement makes company-wide roll-out of new methods possible with a smaller crew of specialists.

1. Detection

The first step needed is to search for the virus. Detection is sometimes an unreliable process which yields false positives. Detectors often use heuristics and fuzzy logic to raise suspicion. For the purposes of FIXALL, it is only necessary to have a suspicion that the virus is present.

FIXALL uses very simple methods for detection. To locate a boot sector virus in RAM, we examine only the top of memory. In the case of file infectors, we find scanning Memory Control Blocks (MCBs), and the memory just beyond the last MCB, to be sufficient. This sufficiency applies only to those viruses currently included in FIXALL.



Lettvin: 'FIXALL is still under development, with the exclusive charter of perfectly and completely removing viruses for which it has data.'

Any method which will detect the viruses in our list of epidemic viruses is acceptable. During detection we do not care about false positives, because the next step is uncompromisingly accurate.

2. Precise Identification

Before removing a virus, it must be precisely identified. To identify a virus, FIXALL decrypts any encrypted code, and accounts for all morphological changes within the sample, ensuring that every piece of the virus code is examined.

The identity of the virus and the *general* name of the infected object must then be displayed. For instance, 'The Ibex virus was found in the Master Boot Record of your fixed disk #1' is preferable to 'Ibex, drive 0x80, sector 1, head 0, cylinder 0.'

3. Acquire Pre-Removal Recovery Data

Some viruses will scramble data unless the virus is present in memory. While the virus is still active, FIXALL collects the unscrambled data for later use. Some viruses encrypt the original Master Boot Sector (MBS) and restore it only when the virus is active. Without precise identification, recovery from premature replacement of the MBS can be difficult.

For file recovery, fetch the original unencrypted contents of file where necessary/possible. Also fetch original file management data when necessary.

4. Disable the Virus

All copies of the virus in RAM must then be located. It is worth noting that some viruses may be active in more than one location. The virus is then deactivated in memory by overwriting the memory-resident copy with code which jumps to the previously chained interrupt vector. It is important that FIXALL does not reinstall any old vectors, as this action may unchain a driver and contribute to data destruction.

5. Report that the Virus has been Disabled

It is now necessary to report that code to bypass the virus functions has been installed and that the operation of the PC will be virus-free until the user engages in the behaviour which led to infection. Usually this means the PC will be virus-free until rebooted from a boot sector virus carrying diskette.

6. Overwrite the Virus in Memory

All virus data other than the disabling jump and data for that jump must now be overwritten. FIXALL uses the NOP instruction as the overwriting data. Every copy of the virus in RAM is overwritten, including all memory reserved by the virus as data space.

7. Restore Original Boot Sector

If the virus has altered the MBS or DOS Boot Sector (DBS), the original boot sector must now be replaced. If the virus stores an unencrypted copy of the boot sector, this copy should be used, otherwise the MBS or DBS should be fetched during acquisition of pre-removal recovery data. If the original boot sector has been destroyed, a temporary replacement boot sector should be created. Additionally, all affected FAT and DIR entries must be repaired.

8. Overwrite all Non-restored Virus-related Disk Areas

Any remaining sectors used by the virus are then overwritten. *Lotus* uses a string naming the virus remover, its version, and the virus it removed. Subsequent recovery efforts will display this string and may help in any investigation of why recovery cannot be complete.

9. Restore Program Files

Some viruses make restoration tricky and ambiguous. Only files for which restoration is guaranteed to be successful should be restored. Users should be informed if restoration cannot be guaranteed, and the suspect file should be backed up with an unusual extension like VOM, VXE or VLL.

The virus code should then be overwritten in the file to be restored with a string naming the virus remover, its version, and the virus it removed. Once again, this information can be used at a later date by an investigator. If the file's header information is altered, it should be restored, where possible. All ambiguously restored files should be wrapped in DOS code, notifying the user of repair efforts, and advising that it is strongly inadvisable to use the file. However, the backup file should be made available in case the wrapped, ambiguous restoration does not function and the user is willing to accept the risks of running the infected file.

10. Document all Infected Files

Every disinfected file should be identified to the user. It is vital that all ambiguous removals are clearly marked as such.

11. Report Virus Removal

The user should then be told that the virus has been removed. Inform them that their PC is entirely free from the effects of the virus if and when this is true. Otherwise, inform them of the precise nature of what damage to expect when recovering.

12. Describe how the Machine became Infected

It is important to give the user a simple, readable description telling them exactly how the virus came to be on their PC. For example, with the Form virus, FIXALL tells users that they left a floppy disk in their boot drive when they tried to reboot. We also inform them that they may have seen the message 'Non-System diskette. Please remove the diskette and hit any key'. Users are then given information on how to avoid the virus in the future.

13. Provide Local Hot-line Information

A user who has just removed a virus must help researchers by providing us with their virus removal information. We need to know where the viruses are and how they are distributed. From the data gathered by the hot-line, we can make good guesses about where to put our efforts.

It is important that the virus removal program permits custom messages to be output to users, so our hot-line can gather this data.

14. Recommend a Reboot

After removing a virus from a PC, there are usually some areas of RAM which are out of use. A reboot recovers these areas of memory. We tell users that this is not necessary but may be a useful precaution.

15. Announce Product Name, Version and Copyright

Circumstances may arise where further output is necessary, although this has not yet been the case. In our experience, too much information during recovery can confuse the user. We simply do not burden the user in this way, and ask other removal program vendors to avoid doing so as well.

16. Speed is Essential

Any virus removal product should take no more than one second to run. Its only task is to provide a virus-free execution space and remove viruses from certain specified files. FIXALL is loaded from AUTOEXEC.BAT, and unless a virus is detected, is transparent.

17. Terminate, but Stay Resident

Infected executable files should be cleaned of epidemic viruses as they are submitted for execution. The resident image can be very small, if properly designed.

18. Pause, and Non-zero Error Level

A pause and non-zero ERRORLEVEL should be provided only when a virus is found. The user must not be able to escape from the pause casually. FIXALL uses the ampersand key, which requires the use of two key presses. The user must read the messages output by FIXALL to learn that this is the only key they can use to quit.

The non-zero ERRORLEVEL allows batch files to respond to prior virus conditions.

Overview and Motivation

We at *Lotus* know of no commercial anti-virus product which fulfils our virus removal criteria. I will use Form as an example. Form can be disabled while it is active, and then removed entirely from the disk with very little effort. However, many anti-virus products cannot do this. These products also only partially remove Form from the disk, leaving scraps of virus code and an offensive message which we find unprofessional. Given the simple nature of the Form virus, and its global spread, this situation is intolerable.

It was this shortcoming which led us to develop FIXALL. Running FIXALL during system startup will instantly and correctly remove a number of viruses. FIXALL will instruct the user of the PC on the behaviour which led to infection. If FIXALL is not able to identify each and every byte of the virus code, it will take no further action.

The *Lotus* virus removal acceptance criteria are completely satisfied by our FIXALL program. Before we accept any other virus removal program for general distribution within *Lotus*, that program must also satisfy the acceptance criteria.

FIXALL is still under development, with the exclusive charter of perfectly and completely removing viruses known to it. Permanent damage done by a virus will not be undone by FIXALL, except in the special case of replacing the Master Boot Sector code when no copy of the original exists, as is necessary with viruses like AntiCMOS. Other special cases may be handled later. FIXALL will never make a PC worse than it already is.

We believe that many independent anti-virus companies are developing good competitive products for detecting viruses. For this reason, a variety of scanners are located in several strategic points throughout the company. However, while focusing on the general virus problem, the special case of the true epidemic virus is not addressed to our satisfaction. That is why I wrote the criteria, and why my colleague Greg Lutz and I continue to develop FIXALL. *Lotus* has no wish to produce commercial anti-virus software, and challenges anti-virus companies to meet these criteria, making development of FIXALL unnecessary.

Conclusions

Lotus has specific virus control needs. As a software development company, we need to be diligent in preventing distribution of viruses to customers. As a software consuming company, we need to allow both experimentation by our employees and, at the same time, prevent epidemic viruses from spreading. As a service company, we need to allow customers to send us diskettes and use other media, while remaining responsible for detecting any viruses which may be sent with that material.

On a larger scale, *Lotus* feels that it should contribute to the global control of viruses as much as possible. Meanwhile, we must strive to improve our understanding of the virus problem, and encourage colleagues to improve theirs.

PRODUCT REVIEW 1

AVP - A Professional Choice

Dr Keith Jackson

Writing my review this month has reminded me how the world has changed in a few short years. When *VB* was first published in 1989, the Berlin Wall still stood, and Russia did not partake in international software development. The situation, like Russia itself, is changing fast: already a small stream of Russian-designed software has begun to arrive in the West - one such package is *AntiViral Toolkit Pro* (*AVP*).

AVP is an anti-virus product, which originated in Russia and was developed by Eugene Kaspersky (who often writes about the internal working of viruses for *VB*). The package is distributed in both 'Shareware' and 'Professional' form. It purports to be 'database oriented professional antiviral software', and comprises a scanner, a database editor (to change what the scanner is seeking), a memory-resident detector, and several utilities.

The scanner includes software to remove viruses from infected programs, facilities to look inside compressed files, heuristic features (called a Code Analyser), and the usual facilities to select various options, and write/view reports. A full on-line hypertext help system is included, as well as on-line descriptions of most viruses.

Documentation

The documentation received with *AVP* was sent on disk as printable ASCII text files. A complete copy of the manual is held in one text file, but beware: this is obviously designed for printing using a non-proportional font. Running off a copy in Times Roman proportional font on my printer produced some very 'interesting' tables and screen dumps.

The manual contains explanations of the various software components, but it has to be said that it is not very easy to read. I am loath to criticise a manual which is probably infinitely better than any of my feeble attempts at communicating in a foreign language, but despite making allowances for the fact that the author is not writing in his mother tongue, there is much ambiguous material, which could prove confusing to the reader.

The task of finding a reference to a particular feature or problem in the manual is hindered by the lack of both an index and page numbers.

My negative comments about the file-based documentation are completely reversed when it comes to the on-line hypertext help system and the on-line virus explanations. These are excellent, in particular the virus explanations, which even include demonstrations of the sound- and screen effects which are generated by many of the viruses known to *AVP*. It is obvious that the developers painstakingly searched through all the viruses, and extracted the code from each one which had such effects - a remarkable effort, the results of which I found entirely fascinating.

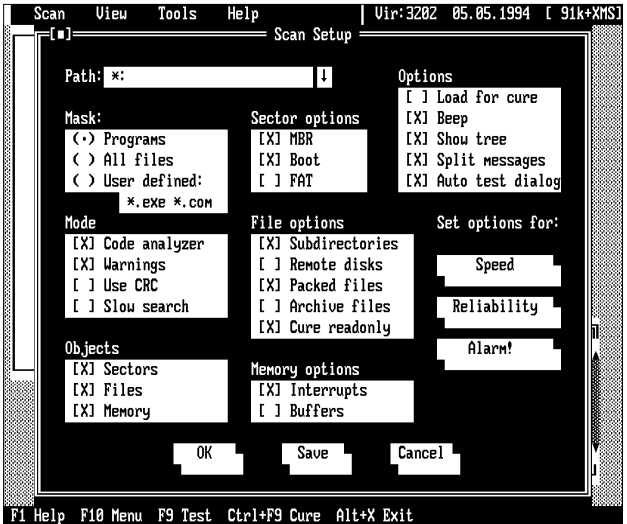
Installation

AVP was provided on a single 3.5-inch (1.44 Mbyte) floppy disk, as shareware. Installation is very easy: after being unpacked it is merely necessary to copy the files to any desired subdirectory. An upgrade of the various databases was provided, and these files are copied across as replacements for the original *AVP* files. Nothing could be simpler. After installation is complete, four executable files are available; the scanner, the 'professional' scanner (which includes the database editor), the memory-resident monitor, and a 'utility' program.

When I ran *AVP*, I was surprised to see an error message stating that some of the database files (specifically EXTRACT.-VB and CA.-VB) were over three months out of date. This occurred despite the fact that the update files which I had installed were only six weeks old. Such things do not help novice users gain confidence in a product.

Scanner Facilities

The scanner used for this review claimed to be capable of detecting 3202 viruses (as of the date of the upgrade files, 5 May 1994). When a scan is performed, a 'scan window' and a 'check-up window' are both visible. The scan window displays a subdirectory tree showing all the non-standard files which have been detected, and marking them appropriately as LZEXE, EXE with COM extension, etc. The check-



Not only is *AVP* accurate, but it has an easy-to-use interface, and is configurable seemingly *ad infinitum*!

up window merely marks files as infected (with a suitable explanation), or 'OK'. By default, the scanner checks carried out a complete check of all available drives, even my magneto-optical drive. All this is obviously fairly time-consuming, but is also very thorough.

The *AVP* scanner offers an excellent range of setup options. In its default (slow) mode it applies heuristic tests, providing warnings as well as definite virus recognition, and looks inside most of the popular compression utilities including DIET, PKLITE, LZEXE and EXEPACK.

Options are also available to alter the types of file inspected (the user may choose between program files only, all files, or a user-defined mask), to inspect various disk sectors, to change the objects which are inspected (files, memory etc.), and to alter the ways in which reports can be produced.

Most helpfully, specific buttons are available to flip the *AVP* setup between Speed and Reliability. This is the main choice which most users will have to make.

Speed and Detection

In default mode, *AVP* took 2 minutes 36 seconds to scan the hard disk of my test computer. However, when the setup options are used for Speed rather than Reliability (the more secure mode), this time was reduced to 44 seconds. Both times are much less than the initial scan of all drives on my test computer, which took 5 minutes 31 seconds. By way of comparison, *Dr. Solomon's Anti-Virus Toolkit* could scan the same hard disk in 20 seconds, and *Sophos' Sweep* took 24 seconds in default (fast) mode and 1 minute 13 seconds for a complete scan.

When tested against the viruses listed in the *Technical Details* section, *AVP* detected all but one - the sole exception being 12_Tricks (a well-known Trojan rather than a virus, and therefore an admissible omission) - an exceptionally good result. When tested against Mutation Engine (MtE) samples, the results were equally impressive: 100% of the MtE-infected test files were correctly detected.

The heuristic part of *AVP* warned that another 56 files could well be infected, by producing various messages that the files were TSR possibilities (i.e. they could Terminate and Stay Resident). This heuristic part produced no false positive results during testing; another excellent result.

When the detection tests were repeated with the product set for Speed instead of Reliability, the same results were obtained, except that the heuristic part of *AVP* did not produce any warnings. This is unsurprising given that it is automatically deselected when the user opts for Speed.

Memory-resident Monitor

The memory-resident monitor provided with *AVP* offers many tailoring options. It is possible to instruct this program to test for access to files, formatting, writing to disk, dangerous calls, and to check for the presence of viruses. All options apart from specifically testing for the presence of a virus are activated by default, making the TSR in essence a behaviour blocker.

The virus-specific option is in fact not particularly efficient, and takes some considerable time to execute. When I activated it, and tried to copy one sample of each of the non boot-sector viruses, only eight viruses from the 149 included in the test-set were prevented from being copied. These were 8888, Butterfly, Datalock, Fish1100, Helloween, Necropolis, Nuke Hard and Starship.

The memory-resident monitor program occupied 15.5K of base memory when detection for viruses was not activated. However, this rose to a whopping 140K when the option was activated. This, or perhaps its dismal performance, could be the reason why it is not activated by default. No reference is made to use of expanded or extended memory.

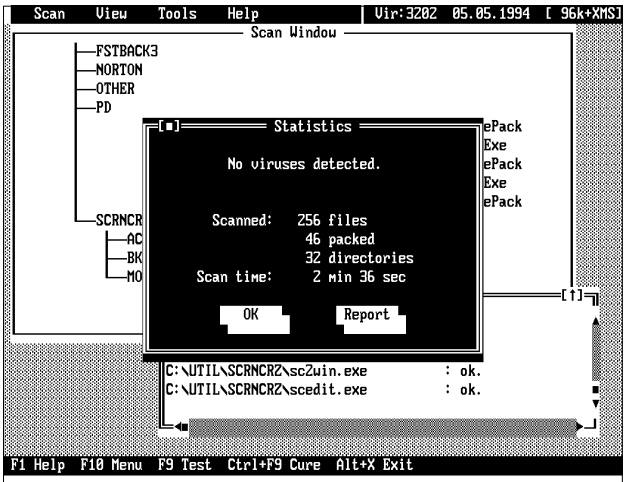
Like many memory-resident anti-virus programs, *AVP's* memory-resident program interferes with routine operation to such an extent that, as far as most users are concerned, its detection capabilities will either be tuned downwards so that interference is minimised, or it will not be used. This inevitably happens when a program is monitoring computer operation for such routinely-used operations as shelling out to DOS in order to activate another program, and deleting or renaming files.

"When tested against the viruses listed in the Technical Details section, AVP detected all but one ... an exceptionally good result"

As an example of such problems, every time I attempted to use my normal scripted procedure to access the *CIX* conferencing system, *AVP's* memory-resident monitor popped up to state that free memory had been reduced and asked for confirmation that this action should be permitted.

This reduction in free memory was caused by my communications program (*Odyssey*) activating its ZModem file transfer protocol program. Unfortunately, the first time this happened, I had forgotten that the memory-resident program was active, and left the room for several minutes. Such things cost money when the telephone call to *CIX* was still active and clocking up units.

AVP's memory-resident program does not like working with *Borland's* pop-up utility, *Sidekick*. If *Sidekick* is activated when the memory-resident program is active, the reduction in free memory queried by *AVP's* memory-resident monitor program is accepted. Then, when the user removes *Sidekick* from memory, the entire screen is covered in a regular pattern of 'snow', and the computer locks up irretrievably. The 'three-fingered salute' (Ctrl-Alt-Del) was necessary to escape from this position.



During every scan, *AVP* keeps the user informed of packed files, and optionally scans their contents.

Other Features

As usual, I have not discussed the package's file virus disinfection features - such attempts always strike me as fundamentally flawed. Although I have heard all the various arguments about supporting users in the field, I remain firmly convinced that such tactics should not be employed, and never review these features.

Infected executable files should be replaced by non-infected files, not by cleaned-up copies. If this proves difficult or impossible, then the capability of the backup system should be thoroughly investigated, as something, somewhere, is badly wrong.

The database system included with the professional version of *AVP* seems to permit complete control over all features of the various database files included with the package. With the exception of a special file called SAMPLES.-VB, every time I tried to activate the database features, a message was displayed saying that the file was locked. Why did this happen? I don't know. Unfortunately I could find no way to circumvent this message, and the manual rather confused me on this point. [*According to KAMI, the bases are locked in order to prevent them being hacked by virus writers. Ed.*]

The utilities included with *AVP* enable the user to look at memory maps, inspect the interrupt vectors, dump various portions of memory, disassemble software into mnemonics, and stay memory-resident. They all seemed to work fairly well under most circumstances, though I admit to a sense of *déjà vu* as far as most of the features are concerned - other utility programs on the market cover much the same ground.

Conclusions

If you are looking for an anti-virus scanner which is excellent at detecting almost every known virus, *AVP* is eminently suitable. Its virus detection capabilities are superb, and it has quite obviously been developed by somebody who knows what he is saying where viruses are concerned.

Having said this, there are still some rough edges in the *AVP* which could do with more refining. The documentation, for example, needs much more work on syntax and structure. Even allowing for the fact that the author is writing in a second language, the time for rewriting has arrived: as mentioned above, there are some places where the lack of clarity introduces confusion and ambiguity.

This applies particularly to the explanation of the database section, which is incomprehensible. All this, however, is eminently solvable, and as *AVP* becomes better known, no doubt the resources will be more readily available to improve this aspect of the product.

Alongside the main scanner, which is excellent (one of the best at detecting viruses, and highly configurable), the utilities and the database rest rather oddly. Their inclusion in the product has rather the feel of an artisan selling his tools. In all probability they have been developed for internal use. They do appear to be a useful addition in marketing terms when they flesh out the product specification.

However, why a user would want to alter the database of virus information is beyond me (developers should do that), and if you want utilities, frankly *Norton* or *PC Tools* offer far more functionality for a pittance. There is nothing much wrong with the inclusion of these components in *AVP*, but the fact is that they simply do not form the core of the product as far as the user is concerned.

I say incessantly that people should use at least two scanners from geographically disparate sources, as the small incestuous world of European and American anti-virus developers has bred a sequence of products all of very similar functionality and capability. One solution to this problem is to find scanners outside the mainstream products which can be recommended. Given its excellent rate of detection, *AVP* is definitely one to try.

Technical Details

Product: *AVP*

Developer/Vendor: Eugene Kaspersky, *KAMI Corp.*, Russia, Tel. +7 095 262 1294, Fax +7 095 270 9418, Email: eugene@kamis.msk.su, Fidonet: 2:5020/156

Availability: Not stated.

Version evaluated: 2.0

Serial number: None visible

Price: US$50 for telephone support from Moscow. Worldwide, prices vary depending on region.

Hardware used: A 33 MHz 486 clone with 4 Mbytes of RAM, one 3.5-inch (1.4 Mbyte) floppy disk drive, one 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 used in the test-sets, see *Virus Bulletin*, February 1994 p.23.

PRODUCT REVIEW 2

NetShield 1.5

Jonathan Burchell

NetShield is the NLM version of *McAfee's* virus scanner. *McAfee* employs an unusual method of distribution, making its products freely available via electronic means and, for a nominal cost, from agents. If the user, after obtaining and installing the software, decides that it fulfils his requirements, he simply sends *McAfee* the licence fee. The fact that the company has survived since 1989 on such 'honesty revenues' implies that many people have been highly satisfied with the *McAfee* offerings: does *NetShield's* performance measure up to its pedigree?

Obtaining the Product

An electronic copy of the product, complete with documentation, can be obtained from the *McAfee* BBS, *CompuServe*, and various other BBSs worldwide. In addition, it is possible, with *Internet* access, to obtain copies of the software via ftp from mcafee.com and various mirror sites.

Alternatively, it can be obtained on floppy disk from any of the *McAfee* agents, complete with a slim printed manual, which describes the software and its operation. The software fits onto a single 3.5-inch, 720K low-density floppy. The manual contains no diagrams or screen shots in the main body of text and only a few screen shots in the appendix; presumably because the main bulk of the documentation is simply a reprint of the text files distributed with the product.

Installation

The printed manual implied that an install routine is on the disk. This is not true: to install the product, one must begin with the readme.1st file on the disk. The actual software and documentation are stored on disk as self-extracting archives. Disk contents are copied to a convenient directory, and several executables run to extract their contents, producing several more READ.ME-type files for the user's perusal.

Some problems now surface. *McAfee* requires that the file server be 3.11 or greater: if it is not, it will first be necessary to apply some *Novell* patches (obtained by extracting the *Novell* supplied patch libraries and installing one or two NLMs). At this point I became nervous - there was little explanation as to why I should apply the patches, and the supplied *Novell* patch libraries contain many more patches than seem to be required by the *McAfee* software.

It is not clear if all patches must be applied (as the *Novell* documentation would have you believe) or just those mentioned by *McAfee*. The exact state in which the 'previously working' file server will be left also begs further explanation. However, I pressed on and installed only the NLMs mentioned by *McAfee* (a newer CLIB.NLM and SPXFIX2.NLM) - all this is done by hand. After patching the server, it is necessary to select either the *NetWare 3* or the *NetWare 4* directory from the temporary installation directory, and extract and copy the files to the server.

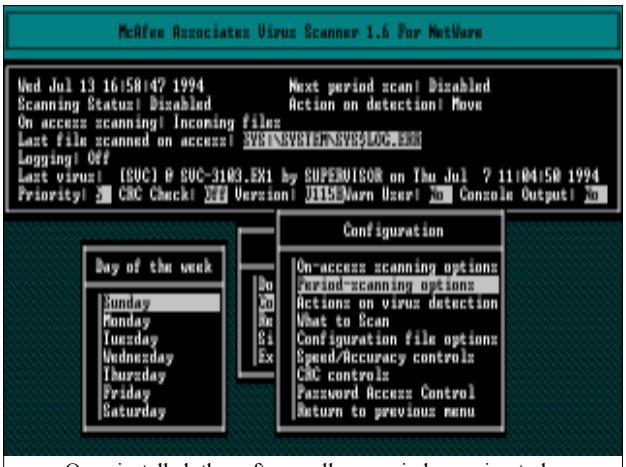
The manual suggests that these files go in the SYSTEM directory: however, this would allow the file server to become a mishmash of *Novell* and third party NLMs. I saw no install option for setting the location of the signature file, so was forced to follow *McAfee* advice. Were I doing this for real, I would certainly investigate whether the software could cope with an install in a subdirectory of SYSTEM. An option does exist for updating the signature file from a file in another location, so it would probably work.

If the installation sounds frightening and distinctly user-unfriendly, that's because it is. It is not particularly difficult, but does leave one with the feeling of charting unexplored regions and hoping 'It's all going to be OK in the end'.

In addition to the limited product documentation, a text file is provided, describing the viruses for which the scanner will check, and giving extremely brief details of each virus' behaviour. No electronic browser of this data or the READ.ME files is provided, and they cannot be regarded as on-line help or an encyclopaedia.

NetShield Features

NetShield provides real-time and background scanning of files on the file server. It provides no support for *Mac* files and has no concept of organising groups of file servers into domains to be configured and administered centrally. In fact, the only recognition of other servers in a network is the ability to have them cross-update signature files so that the newest is automatically copied to all servers.



Once installed, the software allows period scanning to be undertaken. Unfortunately, this option is not as flexible as that of many of its competitors.

No workstation utilities are provided for configuration and administration. This must all be carried out via the main server console screen of *NetShield*, which can be successfully accessed via the *NetWare* remote console, and provides a basic combined status and *NetWare*-type menuing system.

In addition to virus scanning, the NLM is capable of checksumming files and producing an alert when file contents are altered. No details of the exact algorithm are provided, so it is impossible to comment on its usefulness. Options exist to carry out a fast or full CRC check, but the manual implies that the fast check will examine only the start and other critical areas of the file. With certain types of virus, this strategy would be useless.

Administration and Configuration Facilities

On-access scanning options allow selection between real-time scanning of incoming and outgoing files, or of incoming or outgoing files. Real-time scanning can be disabled, and has its own set of 'what to scan' specifications.

Period-scanning is also permitted: a simple-to-use menu allows the user to select daily, weekly or monthly scans, and also the exact time of the scan. Only a single set of 'what to scan' options is provided, so it is not possible to organise different depths of scan at different times (for instance, having a quick scan of highly-used areas every day and a full scan twice a week would not be permitted).

Background scan priority may be set to between one and ten. According to *McAfee*, priority one adds 40-50% CPU loading, whilst priority ten adds only 1-2%. It is not clear whether these figures are correct: perhaps the loading algorithm is simply overcautious. When I set off an immediate scan at the default priority of five, it took just over ten hours to scan some two Gigabytes of files, making this easily the slowest background scanner to date.

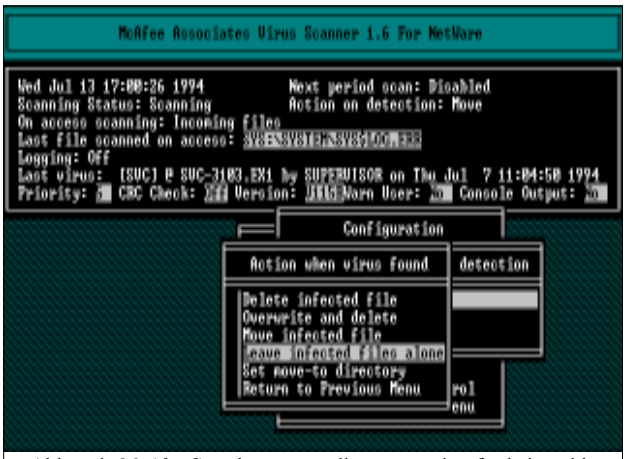
'What to Scan' Options

This list controls real-time and background scanning. The options allow various specifications, such as the file server volumes to be scanned. Real-time scanning always checks all mounted volumes; by default, period-scanning checks all mounted volumes, but can be limited to specific volumes.

There is also an option allowing for particular files to be CRC-checked. Specification of files not to be CRC-checked is also permitted. For this, a complete path must be specified, and wild cards are not allowed.

The default list of extensions for files to be included in the scanning process is COM, EXE, OV? and SYS: BIN and DLL are notable omissions. It is also possible to specify '*', to mean all files. A separate list of extensions is maintained for real-time and periodic scanning.

A single list of extensions not scanned is also maintained. By default, this list is blank, and it is difficult to see what one might put in it. It is possible that this feature might be useful if a file was causing a false positive with the scanner, or if the file server contains a number of static files with an executable extension which are in fact data files.



Although *McAfee Scan* has an excellent reputation for being able to detect viruses, *NetShield* seems to be falling behind, scoring only 90% against the 'In the Wild' test-set.

Specific user's files may be ignored during on-access scanning. The manual suggests that this might be useful to allow unattended users such as a backup process to continue without generating an alert: presumably the rationale is that a backup process is only copying data and so will not execute an infected file and release the virus. Frankly, I find this logic extremely flawed. Apart from the fact that a backup process may well execute files as part of its work, I want to know if I am backing up infected files.

It is possible, with the Skip Directories option, to allow specific directories to be excluded from the scanning process. The suggested use is to prevent the quarantine directory from being scanned - one would think the software could have figured this out for itself.

Finally, the user may choose which action is to be taken on virus detection: both file actions and contact actions may be selected. File actions allow selection between deleting the infected file, overwriting and deleting it (preventing later recovery), moving it to a quarantine directory, or taking no action whatever. Contact actions allow specification of a list of users to be contacted when an infection is found, and the optional disabling of the display of messages on the console. No facilities are provided to change the list of users based on scan type, or to customise the infection message.

Whilst the 'What to scan options' offer some flexibility, configuration is limited. The inability to set differing types of period scan is annoying, as is the inability to say 'Scan only these directories'.

Logging and Reporting

It is possible to have the NLM report its progress, and any events, to a log file. Although the ability to view and print the log file is provided, control of what goes into the file is not, nor is filtering the file contents permitted. If selective

reports of the file generated are required, the code must be written by the user; as the log file is completely undocumented, this will not be a simple task.

A final set of options allows saving of the current configuration and optional loading of a new one. Presumably, this could be used to provide a finer level of control such as different period scans on different days; however, it would at best be a rather messy sludge.

Updates

As mentioned earlier, the product, updates to signature files, and the detector can be obtained electronically. *McAfee* provide two very useful utilities to help with obtaining and verifying updates. *Software Express* is a *DOS* and *Microsoft Windows* program which largely automates the task of logging into the *McAfee* BBS and obtaining the latest software. It is an extremely sophisticated program and has more automation and a slicker interface than the NLM itself!

Having obtained an electronic copy of the software, it is vital to check that it has not been tampered with, a process carried out by the validate program - this confirms that no alterations or Trojans are present. Exactly what the program checks is not explained, so it is impossible to be sure that an advanced hacker would be unable to fake the results.

Results

The scanner turned in a good detection ratio only on the 'Standard' test-set. However, it detected only 98 of the 109 'In the Wild' viruses. Despite the fact that this is almost a 90% detection rate, it is a poor result: all of the viruses in this test-set have been found at 'real world' sites, and are known to be active. It is unacceptable that any anti-virus software package should not detect 100% of these viruses.

Equally disappointing is the low result on the 'Polymorphic' test-set: a detection rate of little more than one-third indicates strongly that more work needs to be done in this area. The developers have no grounds to claim that they had never seen the viruses - in an earlier test (*Virus Bulletin*, March 1993, pp.20-22), *Scan* performed flawlessly against a similar test-set containing 1024 MtE samples. When contacted, *McAfee Associates* pointed out that it recognised these problems, and will address many of the points raised in this review in the forthcoming launch of *NetShield 2.0*.

Conclusions

NetShield is an extremely basic product, lacking many features integrated in other server-based products, and which are now almost *de rigueur*. With improved polymorphic detection ratios it might be worthwhile considering using it in a small network (around 10 machines), single-server environment, but the lack of sophistication is a great concern. Even in such an environment, the user would need to check carefully that none of its missing features were ones which were desirable or, indeed, necessary.

McAfee NetShield

Detection Results (Secure mode):

NLM Scanner

Standard Test-Set[1]	223/229	97.4%
In the Wild Test-Set[2]	98/109	89.9%
Polymorphic Test-Set[3]	157/450	34.9%

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: *NetShield version 1.5*

Developer: *McAfee Associates*, 2710 Walsh Avenue, Suite 200, Santa Clara, California 95051, USA.
Tel. +1 498 988 3832, Fax +1 408 970 9727

UK Distributors: *International Data Systems*, 9/10 Alfred Place, London WC1B 7EB, England.
Tel. +44 71 631 0548, Fax +44 71 581 1466

Price: Price per node. 1-10 nodes: £378; 11- 25 nodes £504; 25-50 nodes £756, 51-75 nodes £1008; 76-1000 nodes £1260. Updates may be downloaded from the *McAfee* BBS at any time, but guaranteed quarterly updates are also available on disk for £75 (annual fee) for up to 75 users, and free for 76+.

Hardware used: Client machine - 33 MHz 486, 200 Mbyte IDE drive, 16 Mbytes RAM. File server - 33 MHz 486, EISA bus, 32 bit caching disk controller, *NetWare 3.11*, 16 Mbytes 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-T2, 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).

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
Email virusbtn@vax.ox.ac.uk
CompuServe 100070,1340@compuserve.com

US subscriptions only:

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

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

END NOTES AND NEWS

According to a report by *Sigma Group International Ltd*, a group of virus writers from Virginia, USA, has begun publicising **the on-line availability of the NuKe library of computer viruses**, a virus construction toolkit, and instructions on how to subvert existing anti-virus programs. As there are no laws in force in the USA banning the creation or distribution of viruses, it is unlikely any formal action can be carried out.

Elsevier Advanced Technology has announced the launch of a new computer security magazine, *Network Security*, from June 1994. Editor John Meyer said: 'The time had now come to devote a single publication to … the threats faced by data transmission in this snowballing area of technology.' Free E-mail news bulletins (*NetSec News*) are also available on a regular basis. Tel. +44 (0)865 843848, E-mail netsec@elsevier.co.uk.

S&S International will be holding **Live Virus Workshops** on 21-22 September, 7-8 November, and 5-6 December 1994. Cost for each two-day session is £680 + VAT, and all workshops will be held at the *Ashridge Management College*, Hertfordshire, UK. Details available from *S&S'* Seminar Department. Tel. +44 (0)295 318700.

PC Guardian has announced the release of *Data Security Plus v5.6*. The product is *Windows 3.1*-compatible, can be centrally administered on a *Novell* network, and provides data protection as well as virus prevention, detection, and removal capabilities.

Software distribution CD-ROM cracked. A CD-ROM containing thousands of pounds worth of encrypted software programs distributed by *Yellow Point* at *CeBIT* has had its password protection cracked by hackers. The CDs were designed to allow users free access to demo versions of software, but required users to pay for decryption information in order to access the full version. Unfortunately, hackers got there first, announcing their triumph during the show.

The *Eleventh World Conference on Computer Security, Audit and Control* (*Compsec 94*) will be held in Westminster, London, UK from 12-14 October 1994. It will incorporate the third annual directors' briefing on computer security, on 13 October. There will also be an exhibition, running concurrently with the main conference. For information, contact Karen Giles on Tel. +44 (0)865 843659, Fax +44 (0)865 843971.

Sophos is holding a **Computer Virus Workshop** at the *Sophos* training suite in Abingdon, near Oxford on 20/21 September, and 23/24 November. Cost for one day is £295 + VAT, and for both days £545 + VAT. Tel. +44 (0)235 559933.

***VSUM* listings for June 1994:** DOS-based scanning products (figures in brackets indicate when that version of the product was first reviewed in *VSUM*): 1. *Command Software's F-Prot Professional 2.13*, 97.1% (9406), 2. *McAfee Associates ViruScan v116*, 97.0% (9406), 3. *Dr Solomon's AVTK v 6.64*, 94.2% (9406), 4. *Sophos' Sweep v2.58*, 92.3% (9403) 5. *IBM Anti-Virus for DOS v1.05*, 87.7% (9406). **NLMs:** 1. *McAfee NetShield 1.6v116*, 95.8% (9406), 2. *Dr Solomon's AVTK v6.64*, 93.7% (9406), 3. *Sophos' Sweep v2.58*, 92.2% (9403), 4. *Command Software's Net-Prot v1.25*, 84.0% (9406).

***Central Point Anti-Virus for NetWare version 2.5* has been launched**, only one month after *Central Point's* merger with *Symantec*. Features include a new and faster scanning engine, and remote management of virus protection. For further information, readers should contact *Symantec*. Tel. +44 (0)628 592222.

ERRATUM: The Boot Sector Test-set as reported at the end of the Comparative Review in the July edition of *Virus Bulletin* (p.23) was incorrect. The correct test-set contains one sample each of BFD-451, Form, JackRipper, Monkey, New_Zealand_2, NoInt, Parity_Boot, Quox, and Spanish_Telecom, not the list of viruses published.