February 1993 ISSN 0956-9979

VIRUS BULLETIN

THE AUTHORITATIVE INTERNATIONAL PUBLICATION
ON COMPUTER VIRUS PREVENTION,
RECOGNITION AND REMOVAL

CONTENTS

EDITORIAL

Who Wants To Buy?

Last month *Virus Bulletin* reported that a man had been arrested for selling malicious computer code. The morals of the case seem to be self evident, but there are still proponents of this trade hiding behind the banner of free speech. As adverts offering virus code or virus creation toolkits become increasingly common, important legal cases are approaching which should finally prove whether selling virus code is illegal in the UK.

Those who sell computer viruses often see themselves as visionaries, bringing coveted knowledge to the masses. However, most anti-virus researchers see these vendors as pariahs - living off the misfortune of others and motivated by money, individuals whose actions only make the situation worse. What response the industry should take is a question which must be answered *now*, as the floodgates may be about to open.

In our society information is, as a general rule, freely available to all. It is behind this high-minded principle that the retailers of malicious code hide.

Mark Ludwig, author of the *Little Black Book of Computer Viruses*, claims that 'modern society is based upon intellectual freedom versus responsibility' and that by effectively selling virus code he is simply exercising his intellectual freedom. How far the idea of 'exercising academic freedom' goes is arguable - does it cover selling small titbits of information, or does it mean releasing a CD-ROM crammed with every virus known to date? Better yet, what about writing some viruses, to make sure those rights really get the exercise they need...

It can be argued that to combat crime effectively, one first needs to understand it. There is little doubt that those in charge of anti-virus policies within organisations need to have worked with a live virus in order to appreciate the associated problems and pitfalls fully. However, this should always be done in carefully controlled conditions which do not involve further distribution of viruses.

To find an adequate solution to the virus problem a solid technical understanding is needed - but not to the extent of writing a new virus to test a principle. An understanding of how viruses work is important, as grasping the implications of new technical developments is impossible without it. However, for virus *protection*, analyses of the form given in *Virus Bulletin* should provide sufficient insight - anything more and it is all too easy to give would-be virus authors dangerous ideas.

It is not illegal to write a virus - as long as the virus never spreads beyond computers owned by the author. It is also not illegal to possess virus code - especially as the majority of people who do possess viruses are blissfully unaware of their misfortune! So is it illegal to sell viruses? The answer to the question seems, unfortunately, to be no.

An analogous argument seems to be whether it is legal to sell Police radar detectors. This is legal within the UK, although *using* such a detector is illegal.

The same argument may well be applicable to virus code: it is illegal to infect someone's computer *deliberately*, but if you are sold a virus and told not to let it execute, has the vendor really committed a criminal offence?

Although the legal situation is less than clear, it is impossible for any normal computer user to disagree that selling viruses is an unsavoury practice. There is no convincing argument for further dissemination of malicious code and virus construction toolkits - this is already accomplished far too well by Virus Exchange Bulletin Boards. However, the morals of the situation are of secondary importance to the simple fact that it is happening.

The effect of readily available sources of malicious code will be to increase the number of different viruses in the wild drastically. While this certainly keeps scanner developers on their toes, the implications for the user - especially the inadequately prepared user - are very serious. If widespread dissemination of virus code is proven in the courts to be legal, then what next? Internet newsgroups called *alt.fan.darkavenger* and *alt.binaries.ibmpc.viruscode*?

Regardless of personal feelings about selling virus code, it is highly likely that this practice is here to stay. As long as there is money to be made there will always be some 'entrepreneur' willing to sail close to the wind. The law may or may not be able to protect the user in the long run, but a sound anti-virus policy certainly will.

To many this may seem unjust, as the innocent suffer while the guilty proceed quite openly with their business. However, in an imperfect world the industry and users alike have no choice but to weather the approaching storm.

If users wish to stop this trade then they must act, by reporting to the appropriate authorities when their computers are damaged by a virus. The greatest enemy of all is apathy: if users are prepared to sit on the sidelines without comment or complaint, they should accept some of the blame for the problems they face today. Society as a whole *may* act to reform the laws on this issue, but in order to do so, it must be informed, clearly and frequently, that there is a real problem.

NEWS

Saturday The 14th...

The latest anti-virus software marketing ploy has caused one or two raised eyebrows at the *Virus Bulletin* office: the *S&S International* Virus Calendar (see *End Notes And News* for details).

A press release entitled 'Calendar Cautions Against Venomous Viruses!' announces the next line of defence for the embattled computer user. This curiosity 'shows the viruses due to trigger on most days of 1993... and warns of even more damaging viruses such as Flip, which is due to hit on February 2nd, and has an infectious rating of 4, and a damage rating of 4.'

Although increasing awareness of the virus problem is a good thing, the industry has long since decided that such 'date watching' is counter-productive at best. There are now so many different trigger conditions that it is completely meaningless to compile a list of trigger days. The Spanish Telecom virus, for example, triggers after a set number of boots which can happen on any day of the year.

No day should be perceived as more dangerous than any other and, as readers of *Virus Bulletin* will be aware, the professional approach to the problem is one of *constant* care and vigilance. Users should be made aware of the general dangers of computer viruses, and not be encouraged to worry about individual trigger dates.

According to the press release, there are 'only 29 virus free days predicted for 1993'. This is nonsense: the only day which can be considered virus free is one when the computer is not switched on ❑

New Virus Construction Kit

Phalcon/Skism has released a new semi-polymorphic virus construction kit. According to its documentation, the kit, named G², is 'the most modern virus creation tool to date', and allows users to create viruses at an alarming rate.

The instructions for the G² kit come with a standard disclaimer which attempts to distance the author, 'Dark Angel', from any responsibility for his work. Although the code generator routines contain no trigger sequences, it is very simple to add a destructive code generator into G².

G² is designed to produce source code suitable for immediate compilation by most popular assemblers. This makes detection somewhat more difficult as different assemblers can produce slightly different executable code.

Virus Prevalence Table - September 1992

Incidents reported to *VB* during September 1992

Virus	Incidents	(%) Reports
Form	16	27.6%
Spanish Trojan	8	13.6%
New Zealand 2	6	10.0%
Tequila	4	6.8%
V-Sign	4	6.8%
Michelangelo	3	5.1%
Cascade	2	3.4%
Dir II	2	3.4%
Vacsina	2	3.4%
4K	1	1.7%
Beijing	1	1.7%
Dark Avenger	1	1.7%
Darth Vader	1	1.7%
Father	1	1.7%
Helloween	1	1.7%
Italian	1	1.7%
Joshi	1	1.7%
Liberty	1	1.7%
NoInt	1	1.7%
Penza	1	1.7%
SBC	1	1.7%
Total	59	100.0%

The properties of G² are summed up in the following extract from its documentation:

```
PURPOSE
G² is NOT a modification of the Phalcon/Skism Mass-
Produced Code generator; it is a complete rewrite
and functions in a radically different manner.
There will be no further releases of the PS-MPC, as
the project represented the amoeba-stage of virus
creation.

G², Phalcon/Skism's newest virus creation tool, is
designed to allow everyone easy access to computer
virus source code. More than a simple Vienna hack
generator, it creates viruses 'on-the-fly' as per
the user specifications. G² is designed to be
easily maintainable and extensible through the use
of special data files created especially for use by
the program.
```

Virus Prevalence Table - October 1992

Incidents reported to *VB* during October 1992

Virus	Incidents	(%) Reports
Form	22	37.9%
New Zealand 2	10	17.2%
Joshi	5	8.6%
Cascade	4	6.9%
Spanish Trojan	3	5.2%
Helloween	2	3.4%
Jerusalem	2	3.4%
Tequila	2	3.4%
1575	1	1.7%
Captain Trips	1	1.7%
Flip	1	1.7%
Keypress	1	1.7%
Macho	1	1.7%
Michelangelo	1	1.7%
NoInt	1	1.7%
Nomenklatura	1	1.7%
Total	58	100%

G² arrives after the Virus Construction Lab and the Phalcon/Skism Mass-Produced Code generator. These two excellent code generators have several shortcomings inherent in their design. First, they only create one specific virus given a specific configuration. Their design allows for no variability in code generation. Second, upgrading the generated code means getting a new COM or EXE file. With the overhead of the IDE code in VCL, this means redownloading over 100K each release, most of which is redundant. Although the PS-MPC is much smaller and certainly better written, it still suffers from the same lack of simple upgrades. The problem arises because the data needed by the programs for code generation are hard coded, and not in easily updated external files.
G², of course, has none of the problems associated with earlier virus generators. Designed with configurability, variability, and upgradability in mind, G² represents the current apex of virus generation.

The danger of G² lies is that its code generation routines are very flexible. The data files which hold the code from which G² will generate its assembly routines are easily updated and added to - a feature which could make it difficult to detect viruses created by G².

The rise in the number of virus creation toolkits is a cause of some concern within the industry - and rightly so.

Malicious code is now relatively easy to obtain, and the possibilities created by software such as G² are not pleasant. Users and anti-virus developers alike now have no choice but to wait and see what the next wave of DIY virus kits will look like ❑

Don't Panic

'Mainframes hit by epidemic of viruses' was the headline run in *Computing*, a UK weekly computer newspaper. The report went on to explain that 'A survey of 816 European and North American mainframe sites showed that 35.5% of users had reported disasters, 60% of which were due to viruses.' However, mainframe sites can rest at ease - the mainframe virus is not upon us.

Nowhere in the article was it made clear that the mainframes themselves had not been damaged by computer viruses. The report examined damage caused to mainframe *sites (including* PCs at those sites) and users should be aware that there is no practical virus threat to mainframe systems at present. Any PCs used as terminals are at risk however, and if executable files are transferred from terminal to terminal via the mainframe it is entirely possible that infection could spread ❑

Virus Prevalence Table - November 1992

Incidents reported to *VB* during November 1992

Virus	Incidents	(%) Reports
Form	12	23.5%
New Zealand 2	8	15.7%
Tequila	8	15.7%
V-Sign	4	7.8%
Keypress 1495	3	5.9%
NoInt	3	5.9%
Helloween	2	3.9%
Italian	2	3.9%
Spanish Trojan	2	3.9%
Cascade	1	2.0%
CMOS1	1	2.0%
Diskspoiler	1	2.0%
Joshi	1	2.0%
Liberty	1	2.0%
Lovechild	1	2.0%
Yankee Doodle	1	2.0%
Total	51	100.0%

IBM PC VIRUSES (UPDATE)

Updates and amendments to the *Virus Bulletin Table of Known IBM PC Viruses* as of 24th January 1993. 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 preferably a dedicated scanner which contains a user-updatable pattern library.

Type Codes

C = Infects COM files **E** = Infects EXE files **D** = Infects DOS Boot Sector (logical sector 0 on disk)

M = Infects Master Boot Sector (Track 0, Head 0, Sector 1) **N** = Not memory-resident

R = Memory-resident after infection **P** = Companion virus **L** = Link virus

Known Viruses

99%.Traveling_Jack - EN: A 980 byte virus, which has not been fully analysed, but includes destructive code (INT 26H writes), as well as the encrypted message: 'ThereCanBeOnlyOne.....Signed - Traveling Jack', where the 'J' is a square-root sign.

```
Traveling Jack    F82E 8C1E 0804 8CC8 8EC0 8ED8 803E 0900 0074 168A 1609 00BB
```

_1689 - CER: A 1689 byte stealth virus. Awaiting analysis.

```
_1689             488E D840 33F6 8944 0180 3C5A 7508 8CC0 488E D8C6 045A 06E8
```

4870_Overwriting.B - CEN: A slightly different version of this unusual virus, which spreads in compressed form. It is not recommended that a search string is used to detect this virus.

ARCV.Friends - CN: A 839 byte virus, written by a person calling himself ICE-9. It activates on April 12th, displaying a message saying: 'released 5th September ARcV Productions Dedicated to all my friends'.

```
Friends           1681 C3?? 1253 5FB9 ??15 81E9 ??12 80B5 ??01 ??47 ???? E2F6
```

Bad - CR: The name of this 389 byte virus is derived from a text string it may display: 'Bad command or file name'.

```
Bad               50BF 2601 B0B8 AA58 ABE8 1200 BA26 01B9 0700 8B1E 2D01 B440
```

Baobab.731 - ER: A short, 731 byte variant of the Baobab virus.

```
Baobab.731        6100 8907 8CC0 8ED8 BA6D 00B8 2125 CD21 2EA1 D702 8ED8 2EA1
```

Black_Jec.Sad.307 - CN: This 307 byte virus contains the text 'Sad virus - 24/8/91',and is detected with the Black_Jec pattern.

Bryansk - CN: A 673 byte virus which activates on Fridays, before 3 pm, making files read-only.

```
Bryansk           4EC6 045C 5783 C72C B940 00FC F3A4 5FB4 2ACD 213C 0575 3BB4
```

Burger.536 - CN: A simple overwriting virus, detected with the Burger pattern.

Chemnitz - ER: A 765 byte virus. Awaiting analysis.

```
Chemnitz          3D00 4B74 092E FF2E B400 ???? ???? 5053 5152 5657 1E06 550E
```

Costeau - ER: A 512 byte virus. Awaiting analysis.

```
Costeau           B8BC 4BCD 218B F20E 1F81 C6A7 00B9 2500 F3A7 74C0 B430 CD21
```

Danish_Tiny.310 - CN: Yet another variant of this virus, with encryption added to the 'basic' virus.

```
DanTiny.310       C65B 01B9 DC00 D1E9 7301 4E8B FEAD 33C3 ABE2 FA5E 595B 58C3
```

EMF.625 - CN: This virus seems to be an improved version of the 404 byte variant, which has been reported earlier. The decryption routine is slightly polymorphic, and cannot be found with a single search string.

Flash.695 - CER: Closely related to the Gyorgyi.749 variant, and detected with the same pattern.

Intruder.1967 - EN: A 1967 byte variant which contains code to format the hard disk. Detected with the Intruder pattern.

Jerusalem.June_13th - CER: A 1530/1535 byte variant detected with the Jerusalem-US pattern.

Keypress.1495 - CER: This new member of the Keypress family 1495 (COM) or 1735 (EXE) bytes long. It has been reported 'in the wild' in USA.

```
Keypress.1495     7405 C707 3B01 F9F5 1FC3 F606 1601 0174 0D8C C005 1000 0106
```

Leprosy.Seneca.392 - EN: Similar to the Seneca variant already known, but somewhat shorter.

```
Seneca.392        80FE 0B74 03EB 1990 B42A CD21 80FA 1974 3EEB 0D90 B42C CD21
```

Leprosy.Surfer - CEN: This 946 byte overwriting virus is based on Leprosy, but it includes an encryption routine very similar to the one used by the Rhythm (808) virus.

```
Silver_Surfer     21E8 0100 C3BB 3001 8A2F 322E 0601 882F 4381 FBE2 047E F1C3
```

Luca - CN: A simple 309 byte virus.

```
Luca              50FE C489 4604 B440 B935 018B D5CD 2190 C646 00E9 8F46 01B8
```

Nygus.752 - CER: Very similar to the variant reported earlier, and detected with the same pattern.

PS-MPC generated viruses: Several new PS-MPC-generated viruses have been found or made available to the anti-virus community recently. They include:

> Abraxas (CEN, 546 bytes), ARCV-1 (EN, 826 bytes), ARCV-2 (EN, 693 or 692 bytes), ARCV-3 (CN, 657 bytes), ARCV-5 (CN, 475 bytes), ARCV-6 (CN, 335 bytes), ARCV-7 (EN, 541 bytes), ARCV-8 (EN, 679 bytes), Eclypse (CEN, 641 bytes), Kersplat (CEN, 670 bytes), Mimic (EN, 2832 bytes), Page (CEN, 696 bytes), Schrunch (CN, 458 bytes), Swansong (CEN 1508 bytes), Walkabout (573 bytes), Z-10 (EN, 704 or 702 bytes).

Suriv_1.B - CR: This variant is 897 bytes long, just like the original, but has been patched slightly, probably in an attempt to avoid detection by certain scanners. It is detected by the Suriv 1.01 pattern. The virus contains the string 'Suriv 4.02', but it is just a very minor variant of Suriv 1.

Tankard - CR: A 556 byte virus, awaiting full analysis. It contains the text 'Tankard vers. 3.01', but to date no other versions have been found.

```
Tankard           80FC FF74 1E80 FC3D 741D 3D00 4B74 183D 006C 7418 80FC 5674
```

Timid - CN: Three new variants of this virus are now known, 290, 297 and 382 bytes long. They are detected by the patterns for the 305 and 306 byte variants which were published earlier.

Tremor - CER: This is a new virus, which has been reported in the wild in Germany. This virus is polymorphic, so a simple hex pattern search is not possible. See *End Notes And News* for further details.

Trivial.37 - CN: Yet another silly overwriting virus.

```
Trivial.37        BA9E 00CD 21B7 4093 BA00 01B1 25CD 21B4 4FEB E32A 2E43 4F4D
```

Trivial.Explode - CN: A simple, overwriting virus, which activates on April or May of any year, displays the text 'Your hard drive is about to explode!'. The virus is 229 bytes long.

```
Explode           BA80 0032 EDB4 05CD 1380 FE20 7404 FEC6 EBEE 80FD 2074 0632
```

Trivial.Wolverine - CN: A 202 byte overwriting virus.

```
Wolverine         B800 3DBA F201 CD21 7271 93B4 3FB9 0200 BAC8 01CD 218B 36C8
```

VCL.DM-92 - CN: A 457 byte overwriting virus, which cannot be detected with a single search string, but should be detected by any program able to detect VCL-generated encryption.

VCL-Chuang - CN: This 877 byte virus is flawed, like many other VCL-generated viruses, and infected programs may cause the machine to hang or reboot. It is detected with the VCL-1 and VCL-2 patterns.

Vienna.W13.679 - CN: This 679 byte variant is very similar to the original W13 variant, but it marks infected files by setting the 'month' field to 15. It contains the text 'written in WARSAW (c) Plumbum'. There is also a very similar 518 byte variant, which contains the same text, a 543 byte variant with a different text message, and a short 450 byte one, which does not contain any text messages. All the variants are detected with the W13 pattern, and might be written by the same author.

Wilbur.B - CN: This version is of the same size as the earlier one, 512 bytes, but has not been fully analysed.

```
Wilbur.B          F7DE 0BF6 7414 32E4 CD1A 8AC2 8BCE F6F1 32C0 86E0 8BF0 46E8
```

Yankee.XPEH.4752 - CER: Yet another member of the XPEH group. Detected with the Yankee pattern.

INSIGHT

Mark Hamilton

Apples Are Not The Only Fruit

Apricot is arguably one of the UK's most successful PC manufacturers which, with a buoyant order book, seems to be riding out the recession better than most. The company designs, develops and manufactures its PCs entirely within the United Kingdom and has just invested several million pounds in a new Surface-Mount Technology line. In the middle of last year, it successfully introduced a new range of products to the highly competitive Japanese market.

At the company's Research and Development Laboratories on the *Birmingham University* campus I talked to David Henretty, the Principal Engineer responsible for virus matters. *Apricot* has been aware of the virus problem for some time, and partly as a result of this *Apricot* PCs have included 'LOC Technology' hardware security. This enables a PC to be configured in such a way that activities such as booting from a floppy disk are prohibited.

Such capabilities form a strong first line defence against viruses - particularly Boot Sector viruses. 'We've gone virtually as far as we can go with our hardware security system; if we were to go much beyond that with hardware, the costs would make us uncompetitive', he said.

I asked Henretty whether he could foresee the day when users would demand counter-virus measures in hardware. 'I find it difficult to see what could be offered in the way of hardware protection and still maintain full compatibility with DOS and applications they have out there', he said.

'It [customer-demanded additional security] would have to be something offered by all the hardware manufacturers and I imagine that *IBM* would have to take a lead in that - the industry is still driven by *IBM*.'

'I think customers will start looking to hardware manufacturers for more secure machines in general because of the amount of corporate data held on PCs and the ever-growing awareness of security. The virus side is one aspect of security but I don't think it will be the driving force behind the need for more security.'

Catchall Policy

I then asked Henretty about *Apricot's* internal virus policy; does the company have safe computing guidelines? 'We have a policy in force which ensures that all software coming into the building has to be scanned and all software released from the building has to be scanned - regardless of whether it is going to a customer, another *Apricot* building or anyone else. We have set up dedicated machines throughout all our buildings specifically for the task of scanning floppy disks for viruses. People are encouraged to scan *everything* they use, whether it is brought in from home, downloaded from bulletin boards or from magazine cover disks. This covers every point of entry to the building and every point of exit' he explained.

'We have a Virus Committee, which was set up about a year and a half ago, which meets on a regular basis [approximately monthly]. It consists of representatives from all the *Apricot* sites and it meets to discuss any incidents that have occurred. It also formulates company policy and is involved in the education of users both in terms of awareness and what to do in the event of a virus attack.'

I wondered whether the company heeded *Virus Bulletin* advice to use more than one scanner: 'Most people in the company use one scanner. If there are any problems indicated, then one of the people on the Committee have a variety of scanners they can use against the suspect disk. We standardised on *McAfee's* scanner product some time ago and have site licences for it, but I personally have *eight to ten* scanners I use.'

Henretty has found that a solid understanding of the problem can help. He explained that there were fewer problems at his site, where there was a higher level of technical expertise available. However, the company's sales and marketing operation, based in Hatfield, where there is a high volume of machines and software entering and leaving the building, is more of a potential problem area.

'I have strong views - and most of the Virus Committee agree - that it should not be a disciplinary offence for people to bring in disks from outside, as it only forces the [virus] problem underground. We want to bring this out into the open: make people feel comfortable about finding a virus and reporting it, rather than feeling guilty about it.' He admitted that it was currently a disciplinary offence for anyone working at the factory to bring in disks of 'unauthorised software', whereas at the R&D facility, there is a freer atmosphere. 'Obviously with anyone who deliberately or maliciously infects a machine, it would be a different matter - fortunately, we haven't found anything like that!'

Return To Sender

Apricot's disk scanning strategy seems sound but, as with all PC manufacturers, there are machines being returned to the company, for example after evaluation by the Press and customers, which could easily be infected. What happens to

them? 'Most machines are shipped to our customers and to our resellers direct from our factory in Scotland and they have a policy in force whereby every machine at the end of the manufacturing line is scanned. Also, whenever machines are brought back, whether dead-on-arrivals or returns [from loan] they are stripped down and one part of that process is the scanning of hard disks. This is where we find most viruses: on hard disks coming back into the company from customers, journalists and various others.' I asked him what viruses are he was seeing most often. 'The most common virus at the moment is Form - it seems to be everywhere.'

Henretty told me that when they discover a virus on a machine returned from a customer - 'most of the people we have spoken to don't know they have a virus' - they are invited to download a copy of *McAfee's* scanner from *Apricot's* BBS and are talked through the process of creating a clean boot-floppy and how to scan their machines: 'It's all a matter of educating the users', he said.

DIY Protection

Was *Apricot* ever tempted to follow in *IBM's* footsteps and write its own anti-virus software? 'Certainly not a scanner, we don't have the resources. A change detector? We haven't given it much consideration. There are a lot of fairly good products out there, it would basically be re-inventing the wheel. There are some avenues we are going down in the future that may require us to talk to people like *Sophos* and other similar companies.'

Henretty maintains good relationships with a number of individuals and companies involved in security issues in general and anti-virus measures in particular. At one stage, he said that the company had expressed considerable interest in a site licence for Jim Bates' *VIS* software, but that when *Total Control* started marketing it commercially, it had failed to follow up *Apricot's* repeated approaches. 'It's good software that always seems to achieve high ratings - Jim has a passion for accuracy', he said.

Caught In The Act

Henretty believes the *Computer Misuse Act* is ineffective: 'No, I don't see that it will work or be seen to work until there have been a fair number of successful prosecutions. Although he wasn't charged under the Act, Dr Joseph Popp was a classic example - The Aids Information Disk. He was extradited by *Scotland Yard* through the FBI and was due to be brought to trial but the whole thing fell apart.'

Henretty believes that the laws governing viruses need to be strengthened: 'Just as it wasn't a crime to sell CB Radios before they were legalised, it was a crime to use one. The same sort of arguments can be used by the defence for anyone who sells computer viruses - it is the same sort of thing.' How would he like to see the law strengthened and what additional provisions could be provided? 'Phew, that's a difficult one! I can't think of any Act or piece of legislation that could prevent it [the spread of viruses] without severely curtailing peoples' civil liberties or privacy. I suppose you could make it a crime to write a virus, but who defines what a virus is? Any sort of legislation you bring in is going to have repercussions on creativity.'

Any company which has to distribute disks is at the sharp end of the virus problem. *Apricot* has put up defences to ensure this can never take place.

'I'd like to see more successful prosecutions, because I think that's the only thing that's going to act as any disincentive to people writing viruses. I don't think we have a large number of virus writers in this country, but we do have a large number of people distributing them.'

Henretty has clear views concerning Fred Cohen and his virus-writing contest announced last year: 'I was utterly appalled. I've read his arguments, some of them seem to hold a bit of water, but I'm totally against this competition that was set up to encourage people to submit new viruses. He argued that it was an opportunity to channel people away from writing malicious viruses and give them a chance to compete. I don't think that would work - who is going to control what happens to the viruses? There is going to be a demand for copies of the virus that wins so that other people can learn from it.' He does not believe that there is any such thing as a benevolent virus.

Apricot seems to have a sensible, yet not rigorously intrusive or obstructive, anti-virus policy in place. As a manufacturer, it takes its responsibilities very seriously and does its best to ensure that existing and potential customers and evaluators never receive a virus - an example other companies would do well to emulate.

VIRUS ANALYSIS 1

Jim Bates

The Girafe And Coffee Shop Viruses

Within the anti-virus industry there have been rumblings about virus naming schemes for several years. This is in spite of the obvious problems of attempting to collate the efforts of various warring factions. Virus names were almost invariably allocated on an arbitrary basis by the first individual to disassemble the code. There are also increasing numbers of viruses which contain an obvious name within their code and although serious researchers dislike using these internal names, the identification for computer users is made easier (particularly if the virus announces itself on screen). For most purposes, the name of a virus is immaterial - its attributes and what it actually *does* are far more important, since these will govern what actions the infected user should take.

The particular viruses under examination here have been named Girafe and Coffee Shop but to compound the confusion it transpires that they are different versions of the same virus (called Coffee Shop). They use the letter checking method to prevent infection of certain files and version 3 (originally called Girafe) contains 15 pairs of letters within the checking routine. These are: CO, SC, CL, VS, NE, HT, TB, VI, FI, GI, RA, FE, MT, BR and IM. These characters are stored within the code as a continuous string of 30 characters and it will be seen that the 10th, 11th and 12th pairs form the word 'GIRAFE'. This must be where the person who named this virus got the name (since there is no other similar reference with the virus code).

However, from here on I shall refer to Girafe as Coffee Shop version 3 for reasons which will become plain. The sample I have of this virus was assembled with the TPE (Trident Polymorphic Engine) module that generates pseudo random encryptions during infection, but for the moment let us examine the actual virus code.

The original Coffee Shop virus sample that I examined did not have encrypted code, neither did it contain any encryption mechanism. However, the trigger display on both viruses uses a series of primitive self-locating and unpacking routines which produce the screen display.

Both versions are very similar and obviously developed from the same code, so for analysis purposes I shall treat them as one. Both versions contain the text 'CoffeeShop' although in slightly different locations (neither of which is accessed within the code).

Attributes

Coffee Shop is a primitive resident parasitic virus which appends its code to COM and EXE files. There is no limit on the size of EXE files which may be infected but COM files must be between 2000 and 53247 bytes (inclusive) in version 1, or between 257 and 53247 bytes (inclusive) in version 3.

During the infection process of EXE files, both versions check for the existence of the NE and ME sub headers (for *Windows* program files) and if either is found, infection is aborted. There is a trigger routine which is invoked after a system date and time check to determine whether the day of the week is Friday and the seconds portion of the system clock is zero.

Installation

When the virus is first executed, the TPE generator code (if present) decrypts and relocates the virus which then proceeds to check the DOS version of the host system. If this is earlier than DOS 3.10, processing returns to the host program and the virus code is not installed. If the DOS version is suitable, an 'Are you there?' call is issued by placing the value 33DAh into the AX register and issuing a DOS function request interrupt (INT 21h). If the virus is resident and active, version 1 returns the value A501h in AX and version 3 returns A503h.

If the virus is not resident, the code checks the current MCB (Memory Control Block) to see whether it can be modified to accommodate the virus. If it can, the modifications are completed and the virus is copied into the new MCB area. Processing then collects the existing INT 21h address and inserts it into the virus code. Next, the address of the virus' own INT 21h handler is hooked into the system interrupt table, thus making the virus code active.

In version 3, processing continues by issuing another special function call. This time the value in AX is 33DBh and examination of the virus code reveals that this function request simply initialises a random number generator inside the TPE section of the code. This call is not present in version 1. Then the system date and time are checked to see whether the trigger conditions are met (see above) and if so, the trigger routine is executed. After the trigger (or if the conditions are not met), the virus code repairs the host file image in memory and processing is passed to it.

Operation

Once resident in memory, this virus permanently intercepts only the DOS interrupt service routine (INT 21h) looking specifically for functions 4B00h (Load & Execute) and 6C00h (Extended Open/Create). There are also intercepts

for special functions used by the virus. These are 33DAh, (the 'Are you there?' call), 33DBh (initialise the TPE random number generator), and 33DCh (execute trigger). In version 1, only the 'Are you there?' function call is present.

The interception of function 6C00h in both versions appears designed to infect files opened with Read Only attributes, but an error in both versions prevents this from operating correctly.

Interception of the Load & Execute function call (4B00h) results in the file extension being verified as COM or EXE (all others are rejected). The header bytes of all accepted files are then checked further for the existence of the MZ header. If found, a further check is made for the NE and ME sub headers in *Windows* executables and programs containing these are rejected. Apart from the difference in the acceptable length of possible COM targets, the infection method is similar in both versions and consists of the predictable appending of virus code, with the related modification of the initial bytes or EXE header contents.

Another difference occurs within the self-recognition code that the viruses use to check whether EXE target files are already infected. In version 1, the header checksum field contains 4021h, whilst in version 3 the sum of the two bytes of the header checksum field is 40h. Recognition of infection in COM files is the same for both versions and consists of testing the high bit of the first byte. This is quite common even in uninfected files and can result from a variety of coding options. It is therefore quite likely that the COM infection will simply not take place in the majority of cases.

Trigger routine

The trigger routine displays the message - 'LEGALIZE CANNABIS' in red, white and blue block characters together with a block graphic representation of a cannabis leaf in green. The code then waits for a keystroke before allowing processing to continue. There are no deliberately destructive routines within either version but several bugs in the code can cause program or system malfunction.

Encryption Modules

The distribution of modular encryption and randomisation routines is a recent development begun by the Dark Avenger. In his case the reason was probably to boost a reputation depressed by reports that his viruses were well down the penetration ratings. Another individual or group has now become hooked on this particular ego trip, and the Trident Polymorphic Engine (TPE) has appeared on various virus exchange bulletin boards. The sample of Coffee Shop (version 3) described above uses an early version of TPE and I now have samples of versions described as 1.1, 1.2 and 1.3. The different versions appear to fix various bugs (of which there are many) rather than improving the application of the code.

An analysis of the TPE code in the distributed modules and in the Coffee Shop virus highlights some interesting points.

Firstly, the version used in Coffee Shop 3 appears to predate version 1.1 and since the documentation insists that version 1.1 'is the first public available version of TPE' it seems likely that whoever wrote Coffee Shop also wrote the TPE. Careful analysis of the code in the five programs under examination reveals that two different assemblers were used: Assembler 1 was used for Coffee Shop version 1 and 3, and TPE version 1.0 and 1.2. Assembler 2 was used for TPE versions 1.1 and 1.3.

The probability is that two different people were working together on the development, each using his own assembler.

Secondly, the length of the TPE module varies from version to version as follows:

- Version 1.0 (from Coffee Shop 3) is 1349 bytes long.
- Version 1.1 is 1378 bytes long.
- Version 1.2 is 1355 bytes long.
- Version 1.3 is 1411 bytes long.

It should be noted that the extra length added to a virus under actual infection conditions by inclusion of these modules will contain random elements because of the operation of the module.

The TPE itself is distributed in the form of an object code module for inclusion within the virus code and allows even the most ungifted virus author to produce a self-encrypting and randomising virus. Like the Mutation Engine, it comes with a documentation file explaining its use and the various options that are available. A cursory examination of the modules shows them to be similar in concept to the Mutation Engine although the end result does display marked differences. As is usual with these attempts at more advanced programming, the reliability of an infected system is seriously compromised.

It is not known at the moment exactly where the TPE module originates from but the phrase 'Amsterdam = COFFEESHOP!' appears as plain text within the decrypted version 3 code. This may indicate European involvement but the global interconnection of virus sources and exchange groups makes such speculation difficult. Within the documentation, the TPE is claimed to have been written by one 'Masud Khafir' and pays obeisance to the Mutation Engine and its author - 'I want to thank the Dark Avenger from Bulgaria for his nice 'Mutation Engine' program. This

fine program has been a great source of inspiration for the TPE'. Such knee-bending and forelock tugging is becoming common. There was even a recent published reference to 'the brilliant mind' of the Dark Avenger. This an obvious manifestation of the twisted mentality displayed by virus writers but such comments within the press serve only to perpetuate their folk hero status.

Although the encryption modules may be included as an integral part of the assembled virus code, their presence should not cause virus researchers to consider this a new virus. Certainly detection methods may need modification, but the concerns of the user should be paramount and since the virus attributes (trigger type and conditions, infection paths etc) remain the same, any virus which uses a multiple encryption module should be analysed and reported as if the module was *not* present.

COFFEE SHOP

Aliases:	Girafe
Type:	Resident, Parasitic file infector.
Infection:	COM and EXE files.
Recognition:	
File	Differs between versions. See text.

Hex Pattern	Version 3 of the Coffee Shop virus is encrypted, and so no search pattern is possible. For version 1 only:
	`6F66 6665 6553 686F 7020 B003`
	`CF9C 3DDA 3375 05B8 01A5 9DCF`

System	'Are you there?' call - Place 33DAh in AX and after an INT 21h AX will contain A501h for version 1 and A503h for version 3.
Intercepts:	INT 21h for infection INT 24h for internal error handling.
Trigger:	On Fridays, if the seconds field of the system clock is 0 when an infected file is executed, the words 'LEGALIZE CANNABIS' are displayed with a picture of a cannabis leaf.
Removal:	Specific and generic disinfection is possible. Under clean system conditions, identify and replace infected files.

VIRUS ANALYSIS 2

James Beckett

Spanish Telecom - Galicia Style

It is now over two years since the world saw the introduction of the first Spanish Telecom virus, a very destructive and infectious program with a mildly political message. The writers chose to use this medium to make known their feelings about the Spanish telephone service providers. 'Lower Tariffs, more services' was the cry... and the author(s) of the virus ensured the advertisement of their message by incorporating a trigger routine which completely overwrites the data on a PC fixed disk.

A variant was seen about a year later, which required minor changes to some scanner programs. 'Spanish Telecom 3' has been named as a continuation of the Spanish Telecom series largely because of the message contained within it, but it does not bear any real resemblance to the Spanish Telecom 1 and 2 viruses.

The sample received consists only of a boot sector infection, whereas the original Spanish Telecom variants are multipartite. The boot sectors are themselves viruses and spread independently, so we may later see a file infection which produces this boot virus.

Spanish Telecom 3 also differs in the trigger method, so either it has been written by an independent author or it is a complete re-write.

Analysis

Any IBM PC compatible will be susceptible to a pure boot sector virus, whether running DOS, *Windows*, *OS/2* or UNIX. However, anything other than DOS (or *Windows*, which still uses DOS services) will probably provide its own device drivers from then on and bypass the virus, so it would not spread any further in these environments.

At boot time such a virus installs itself taking some of the top of memory from the BIOS. It then intercepts the direct disk access functions provided by the PC BIOS and monitors disk accesses made by programs later. In this way, the virus is notified of further disks used in the machine and copies its code to them.

The constraint of the 512-byte sector size of the standard PC limits the complexity of a virus, and most boot sector viruses avoid this by having some of their code stored in the boot sector and the remainder elsewhere on the disk.

Cramming a fully operational virus into the size of a single disk sector allows for only very simple creations, and only a few such as New Zealand and Michelangelo are organised this way.

Form and Italian each require two sectors, while Tequila, Joshi and Flip take six. The original Spanish Telecom boot sector virus requires two sectors.

The Spanish Telecom 3 virus, unlike its nominal predecessors, is contained entirely within the one boot sector which usurps the original. On loading, a simple decryption routine is run, involving a trivial XORing of the remaining code with the constant FFFFh.

INT 13h is then trapped and the virus chains itself to the original handler, watching the system for disk requests.

Infection Method

If a program or DOS itself accesses the most recently-read drive the virus allows the request to go through unmolested; clearly this cuts down the number of unnecessary infection attempts. Initially the boot drive is given this honour - as the virus must have loaded from the boot drive, this must already be infected. It could have avoided the boot disk entirely but that would prevent it from infecting on a floppy-only system. Putting in the code to distinguish the two would push it over the 1-sector boundary.

If the program is requesting a read of a floppy boot sector or the fixed disk MBR, the sector in question will have the virus code copied to it, saving the BIOS Parameter Block or partition table as required. A signature 'V1' is used within the sector to avoid multiple infection, which would lose the original boot sector and render the machine unbootable.

The original sector is stored at head 1, track 0, sector 3 on a floppy and at head 0, track 0, sector 6 on a fixed disk.

Another difference from the previous Telecom strains is that they are stealth viruses, faking the original contents of the boot sector when any program tries to read it. Spanish Telecom 3 does not do this.

Trigger

The trigger condition is tested before infection is attempted: Any disk read during the hour of 12pm on 22 May will print the '¡Galicia contra telefonica!' message on the screen, and format track 1 of the disk before hanging the PC.

Another difference from the previous Spanish Telecom viruses is that they both trigger on the 400th boot of the machine after infection, and do a lot more damage, overwriting all the data on both fixed disks.

If Spanish Telecom 3 triggers, it will leave plenty of information for a data recovery service to work with. Depending on the particular disk geometry, one copy of the FAT may be intact, or may be recovered from two partially-damaged copies. The root directory and the data may also be intact in their entirety.

PS/2 disks, naturally enough, tend to use idiosyncratic mappings of heads, cylinders and sectors. Here it is most likely that data will be damaged, while the FAT and the root directory remain safe.

Origins

Galicia is an area in the North-west corner of Spain. The previous Telecom viruses include a message that they were programmed in Barcelona, a mere 900 kilometres away. The whole thing looks more like a copycat virus than a modification of the previous ones.

The full message was only present in the parasitic section of the two previous Telecom viruses, so if a parasitic 'dropper' of this boot sector exists, there may be more information within it.

SPANISH TELECOM 3

Aliases:	None Known
Type:	Resident Master Boot Sector.
Recognition:	
Boot Sector	Text 'V1' at offset 01B3h.
Hex Pattern	Positioned at offset 22 in the Master Boot Sector: E802 00EB 150E 1FBB 3C7C 8B07 35FF FF89 0743 4381 FB5A 7D72
System	INT 13h vector offset will be 7CB8 on last handler before BIOS.
Intercepts:	INT 13h for infection.
Trigger:	Displays text message '¡Galicia contra telefonica!' and formats track 1 of hard drive.
Removal:	Specific and Generic removal is possible. Restore original Master Boot Sector or use FDISK/MBR under DOS 5 under clean system conditions.

VIRUS ANALYSIS 3

Eugene Kaspersky

Todor - Polymorphic And In The Wild

Many of the most elaborate viruses written to date have come from Bulgaria. Although this new sample has the same highly questionable pedigree as some of its siblings, it is significantly simpler. It does, however, have the dubious honour of being found in the wild.

The virus takes its name from an encrypted text string held within the body of the virus - '(C)Todor', which is presumably a reference to Todor Todorov and his notorious virus exchange bulletin board.

The virus is not memory-resident, and infects only COM and EXE files. The most noteworthy characteristic of the virus, other than it being in the wild, is that it uses a new polymorphic algorithm in an attempt to make it more difficult to detect. The algorithm is not particularly advanced or powerful - the Mutation Engine, for example, is much more complex - it is simply different from those observed to date. It is interesting (and somewhat worrying) to note the rapid increase in the number of polymorphic viruses in the last few months.

A Brief Guided Tour

When an infected program is executed, control is passed to the virus code. The majority of the virus body is encrypted, so its first action is to decrypt the code.

The encryption method used is very simple, and is based on the most common virus encryption technique - the XOR function. This routine has been slightly added to by the inclusion of a routine which alters the decryption key between every word.

The virus hooks INT 24h (the DOS Critical Error Handler) to prevent any spurious error messages alerting the user to its presence, and then starts it infection routine. The first file the virus attempts to infect is the file specified as the command interpreter. It does this by reading the file name stored in the environment string by the 'COMSPEC=' qualifier. For most machines this is COMMAND.COM.

The Features

In a further attempt to avoid errors the virus checks to see whether the disk it is attempting to infect is write protected. It does this by creating and immediately deleting a temporary file in its target directory. If any errors are encountered during this process the virus assumes that the disk is write protected in some way, and does not attempt to infect any files stored there.

The virus preserves all but the seconds field of the host file's attributes on infection. This is not usually apparent when examining a directory listing, as DOS only displays its times in hours and minutes - only the *Windows* File Manager displays this field. Infected files have their seconds field set to 22, and the virus uses this seconds field to determine whether a file is already infected. While this will undoubtedly cause it to exclude uninfected files, false positives are of limited concern to the virus author.

Before infection takes place the host file's header is examined. If the header contains the string '(C)Todor' at file offset 32, the virus skips this file and does not infect it. However, because the body of infected files is encrypted, infected files do not contain this string. This is therefore either left over from a development stage, or has been included to stop the virus infecting files on the virus author's development PC. If the header indicates that the file is compressed with LZEXE, it is not infected.

In addition to the checks outlined above, the virus will only infect COM files whose length is greater than 11000 bytes. EXE files deemed suitable for infection are those which do not contain internal overlay code or data. The criteria the virus uses to determine this is that the length of the EXE file must be equal to the length of the EXE file memory requirements as calculated from the header information.

The next step in the infection process is to modify the start of the host file. The virus changes the EXE header fields which contain the initial values of the CS, IP, SS, SP registers, so that the new EXE start address points to the virus code. The virus writes several assembler instructions into the start of COM files in order to pass the control to the virus when the infected file is executed. These instructions vary, but are of the form of moving the offset of the start of the virus code into a register, pushing that value to the stack, and then immediately issuing a RET instruction.

Destructive Trigger Sequence

Another feature of the virus is its destructive trigger. The virus checks the system date and on every month after August (inclusive) and on every day after 15th (inclusive) it encrypts a randomly selected logical sector of the current logical drive. The encryption algorithm used is relatively simple, but the probability that this sector is the FAT sector or a sector in the root directory is very high. This makes the virus very dangerous, because it can destroy much information with this simple encryption routine. However, the virus

author has used the old INT 25h, INT 26h format for accessing the disk, so this technique will not work on logical drives larger than 32 Mbytes.

The Polymorphic Encryption Routine

The en/decryption used by this virus is simple: each word of the virus code is XORed with the encryption key and then this key is ROLed/RORed once. Whilst the algorithm itself is not complex, the analysis of the virus is difficult as the decryption routine which is generated is polymorphic. The routine which does the encryption can be divided into three distinct blocks: the PUSH block, the setup block and the encryption block.

The PUSH block pushes onto the stack seven machine registers: AX, BX, CX, DX, DI, DS, ES. However those seven PUSH instruction (seven opcode bytes) are separated by 'dummy' assembler instructions like NOP, CLC, STD etc. In order for the virus to transfer control successfully to the host program, these register values must be popped off the stack in reverse order. This is done by substituting PUSHes for POPs in this routine.

The register loading block sets up various registers for use by the virus. The instructions used to do this are variable. One of these variations contains a bug which will cause unpredictable results. This point is addressed below.

The opcodes of the encryption block are also variable. Neither the choice of registers nor the instructions used within the encryption routine will be the same for different infections. The number of possible variants of the decryption routine (without taking into account the number of possible keys) is about 2.3×10^{15}.

Even though the author of the Todor virus has attempted to make life difficult for scanner writers, the virus is not particularly hard to detect. This can be done by exploiting certain structures within the virus which remain unchanged.

Bugs

Practically all computer programs contain bugs and, as a rule, virus code seems to contain far more than its fair share. In this case I found several programming errors during analysis.

The most striking bug was found in the algorithm for COM file infection. The virus checks the COM file length before appending to the file, but it does this incorrectly. As a result, the length of an infected COM file can be greater than 64K. Running this file leads to unpredictable results. Within the virus code is a routine which will prevent this corruption of COM files, but it is never called.

The second bug is in the decryption routine. Approximately one time in every six a superfluous POP instruction is inserted. As a result of this, some infected files will cause the machine to crash.

The virus is also unable to determine successfully whether a file is a COM or an EXE file. The virus examines the file extension before infection but does not check the internal structure of the file. As a result, COM files with an EXE extension will be infected as EXE files, and EXE files with a COM extension will be infected as COM files. These files will almost certainly cause the computer to hang when the host program takes control.

Another of the many errors in the code is that the virus modifies the file beginning and then appends the decryption routine and encrypted code to the file. If a write error occurs during this procedure, the file will be damaged irreparably, as the host file will pass execution to a point beyond the its end. This list of errors is in no way exhaustive, and is supplied simply to illustrate the multitude of problems a virus can unintentionally cause on an infected machine.

TODOR

Aliases:	None known.
Type:	Non Resident Appending Parasitic, Polymorphic.
Infection:	COM and EXE files only.
Recognition:	
Files	Seconds field of the file creation time is 22.
System	No system recognition, as the virus is non-resident.
Hex Pattern	No search pattern is possible.
Intercepts:	INT 24h for internal masking of error conditions.
Trigger:	Encrypts a randomly selected logical sector of the current logical drive by using INT 25h / INT 26h Absolute Disk Read/Write DOS Interrupts. Some infected files cause the PC to hang.
Removal:	Under clean system conditions identify and replace infected files.

TUTORIAL

Fridrik Skulason

File Virus Disinfection: Techniques And Problems

There is little doubt that the optimal way to recover from a file-virus infection is to restore all damaged files from a clean backup. However, although this is the safest method, it may not be practical, simply because the backups might be incomplete, out of date or even non-existent. Even when complete backups exist, many users prefer the simplicity of letting the anti-virus software do the work for them.

For this simple reason disinfection programs are popular. They are not problem-free, of course, but they can be a valuable tool when properly used.

The ultimate objective of a disinfection program is to restore the infected file to *exactly* the same the state it was before infection, which is impossible for some viruses but relatively easy for others. In some cases the virus-infected program can only be restored to a slightly different form than it had before infection, but this problem will be discussed in more detail later.

Disinfection techniques can be divided into two groups - specific and generic approaches.

PART I - Virus-specific Disinfection

A virus-specific approach involves recognizing each virus individually, and storing some critical information about it - in particular how long it is and how it modifies the host file. That information is then used to remove the virus 'surgically', in the hope of leaving the original program intact.

The major drawback with this approach is that it cannot handle brand new viruses. In order to remove a new virus the program must be updated.

The primary requirement of a program of this type is that it must be able to identify the viruses it attempts to remove. This sounds relatively simple, but in practice it seems quite hard to fulfil. As an example of what can go wrong, consider the 'Freddy' virus, which one program misidentified as a Jerusalem variant. Attempting to remove the virus as a regular Jerusalem-family virus destroyed the original program, as this virus appends itself to COM files, instead of placing itself in front of the file, as Jerusalem does. This means that an entirely different disinfection algorithm should have been used.

Problems like this can easily be avoided, simply by using exact identification, for example by taking a checksum of all the non-variable bytes in the virus, and verifying that the checksum matches.

Unfortunately, this approach is not entirely problem-free either. Assume that a new virus is created, by taking an old virus and modifying a single letter in an unused text string inside it. The question is then: If a virus-specific disinfection program can remove the original virus, is it reasonable to expect it to be able to remove the new variant as well?

One group of researchers might argue that the answer is 'No' - this is a new virus, and the program should not attempt do disinfect anything it does not recognize.

Other researchers might point out that the new variant is practically identical to the original one, and that it can be removed in *exactly* the same way as the original, so there is no reason not to do so.

Personally I favour what I call 'nearly-exact' identification - which will identify significant differences between variants, such as when the size changes, or when the location of critical data areas within the virus changes, but does not always notice minor changes, such as different text messages or one-bit random changes in the code.

Assuming the disinfection program is capable of correctly identifying the virus, the next step is to select the disinfection method to use. As program viruses infect in several different ways, from a structural point of view, a disinfection program of this type must have a corresponding set of methods. The number of different infection methods is a matter of definition, but it is probably between 15 and 20.

Only a few of those methods will be discussed here, as the differences between them are sometimes rather small.

Group 1: Companion Or 'Spawning' Viruses

Disinfecting these viruses is extremely easy - in fact, all that needs to be done is to delete the file containing the virus, as the original file has not been modified at all.

Group 2: Overwriting Viruses

This is also a very simple case - overwriting viruses by definition cannot be disinfected, unless the disinfection program has previously saved information about what the overwritten part of the program looked like - something which the virus-specific programs do not do.

Group 3: Simple COM-appending Viruses

The largest group (over 500) contains the viruses which simply add the virus code to the end of the file, and place a

JMP or CALL at the beginning of the program. In this case the disinfection process consists of two steps - locating the few bytes which were overwritten, moving them to the beginning of the file, and finally truncating the file to its original size.

Group 4: Simple EXE-appending Files

As EXE files are somewhat more complex structurally than COM files, it is not surprising that disinfection is slightly more involved than in the previous group. The virus modifies certain fields in the file header, and the methods used to restore them are somewhat variable. In some cases the original header is stored complete within the virus body, but in other cases the virus only stores the original value of the fields it changes. Several viruses use even more complex methods, such as increasing certain fields, (eg the initial SS value) by a fixed number when infecting.

Many EXE-infecting viruses make irreversible changes to the header, such as setting the 'checksum' value to a fixed number, to mark the file as infected. Even if the rest of the header can be restored, the resulting program will not be identical to the original one because of the different checksum. Whether this change matters or not will be discussed later.

There are several groups of viruses which are of about the same complexity as the last one, but will not be discussed here. They include viruses which infect SYS files, add their code to the front of files, overwrite the beginning and move the original code to the end, and a few other cases where the only problem in disinfection is to figure out how many bytes to move from where to where within the files, how large the block of virus code is, and where it is located.

Instead, I will look briefly at some of the more interesting groups, most of which are represented by a single virus.

DIR-II Method

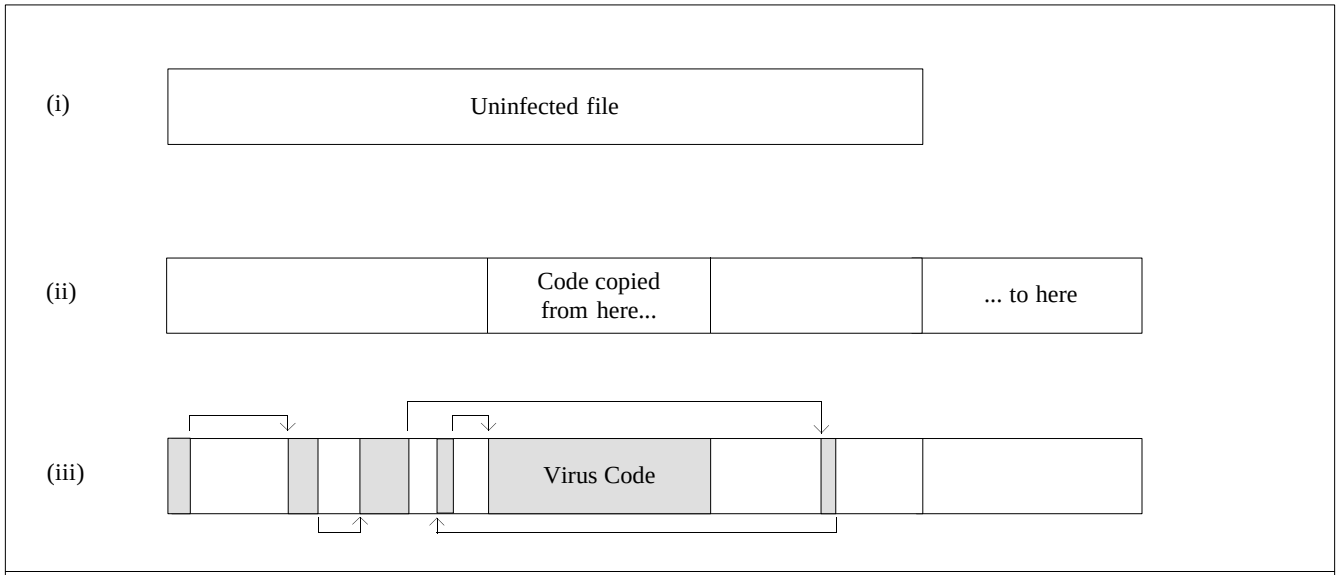
The DIR-II virus is special as it does not really infect files, but rather directory entries, modifying the 'first cluster' field. This virus can be removed by restoring the directory entry itself, but another disinfection method is also possible. The second method requires that the virus be active, and involves a renaming trick, but it will not be discussed in detail here. [*See VB, November 1991, p.11. Ed.*]

0-block Method

A virus can infect a file without changing the size of the host, by locating a block of constants, usually 0, and overwriting it. When the virus-infected program is run, the virus clears the block before returning control to the host.

Commander Bomber

This virus is unusually difficult to disinfect, as the virus body is located inside the file, not at the beginning or at the end. Additionally, the virus may have left small code fragments scattered through the code (see diagram).



The Commander Bomber virus in action: (i) The uninfected host file. (ii) A block of code is copied from the middle of the host file to the end. (iii) The path of execution for a fully infected file.

Virus authors do not only keep coming up with new infection techniques, but they also have a number of tricks up their sleeve - tricks which are designed only to make life more difficult for developers of disinfection software.

These tricks include confusing the identification process by embedding a known signature of an old virus in a new, unrelated virus. One of the Gotcha viruses does this, thoroughly confusing any scanner which uses those patterns. Of course, this problem can be avoided by using exact (or nearly-exact) identification.

Encryption makes disinfection slightly more complicated, and can be quite annoying, for example if the virus uses multiple levels of encryption. Writing a disinfector for an encrypted virus generally takes several times as long as for a non-encrypted one.

PART II - Generic Disinfection

Implementing a bullet-proof generic disinfector is quite hard, but currently two somewhat different approaches seem to be used.

The first method involves either emulating the execution of the infected program, or single-stepping through the code, attempting to discover when the virus returns control to the original program. The idea is that by observing how the virus restores the program, the anti-virus program can restore it the same way. The virus authors have generally not responded to this approach yet - it is rather recent, and not widely used.

Nevertheless there are several different approaches which viruses can take, and probably will take in the future, if these methods get popular. The most obvious one is to attempt to determine if the virus is being run normally or not, and behave in a different way, if it seems to be emulated or single-stepped. The single-stepping idea has not been developed very far, and may not be practical under all circumstances, but it has two significant advantages - it is as effective against old viruses as new and it does not require any prior information about the infected program.

The second generic approach is already used by certain integrity-checking programs. It involves building a database of information about the programs, when they are known to be in a 'clean' state. At an absolute minimum, three fields are necessary:

- The original size of the program.
- A checksum for the program, typically a 32-bit value.
- The contents of the original first X bytes of the program, with X around 30 or so.

Other information, such as the original time/date of the file might be stored as well.

The checksum for the file is then regularly re-calculated, and some action taken if it does not match. The disinfection routine assumes that the changes have been caused by a virus which made a specific type of changes, and attempts to reverse those changes.

For example, if the infected program has grown by X bytes, and the beginning is different from the stored beginning, the disinfection program might try to truncate the file, shortening it by X bytes, and then overwrite the beginning with the stored code.

The last step of this process is that the file's checksum is re-calculated. If the checksum of the 'disinfected' file and the checksum of the original uninfected file do not match the disinfection routine is considered a failure.

The biggest benefit of this approach is that it guarantees that the disinfection will never damage or corrupt the file - that is, it will either give up, or restore the file exactly to its original form.

Like the other disinfection techniques discussed here, this one can be attacked as well. It does not matter if virus is encrypted, but the virus can easily encrypt a part (or the whole) of the original program. A virus-specific disinfector can handle that, but this approach would be defeated.

Maybe this point illustrates why no single approach is dominant today, as virus-specific and generic disinfection techniques might be viewed as complementary, rather than simply competitive.

With all of the above in mind, I would like to leave the reader with two questions, provided as food for thought:

„ Should a virus-specific disinfection program attempt to disinfect all known viruses, or just the most common ones, that is the 100 or so that have any significant chances of ever hitting a 'real user'?

„ Is exact restoration absolutely necessary? Many viruses 'pad' files before infection - making their length a fixed multiple of 16 bytes, so the virus code can start on a paragraph boundary. If the virus is removed, but those extra bytes left attached, the program will almost always work without problems - the only significant exception being programs that perform some internal self-integrity check - even just a minimal one such as checking their own length. Therefore there are certainly cases where it is possible to disinfect a file, but the 'before infection' and 'after infection' checksums will not match.

PRODUCT REVIEW 1

Mark Hamilton

Search... And Destroy

Search and Destroy is a relatively new product on the market from *Fifth Generation Systems.* Like its elder brother *Untouchable*, *Search and Destroy* is developed by the Israeli company *BRM Technologies*. It is deliberately positioned as a junior product to *Untouchable* and whilst some of its functionality is complementary to it, the two overlap to quite a degree.

The software is available on either 3.5 or 5.25-inch disks and accompanied by a 94 page A5 manual and a 24 page Virus Backgrounder booklet. This latter document paints a gloomy, sometimes inaccurate picture. For example:

'First and foremost, scanners are completely ineffective against any virus whose code pattern is not recognised. In other words, scanners cannot identify a virus if they don't have a signature for it.'

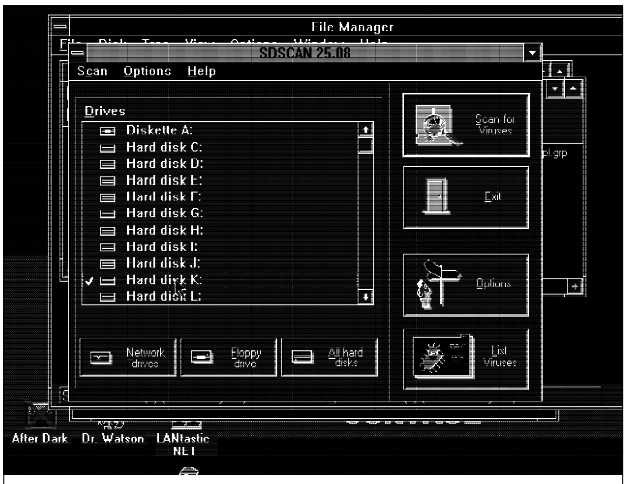
So far, so good, but later in the same paragraph: 'A new wave in virus authorship is the creation of self-mutating viruses. These viruses infect a file in a different way each time, so it cannot be identified by a simple pattern search, rendering virus scanners ineffective.' This does seem a rather disheartened comment, especially when scanners - *Search and Destroy* included - are entirely capable of detecting polymorphic viruses.

Component Parts

Search and Destroy basically comprises of a virus scanner and a virus-specific memory-resident monitoring program. The software is supplied for both DOS and *Windows*.

The DOS TSR program, SDRES, checks each new floppy introduced to the PC and scans each executable program that is loaded and executed. If expanded memory is available, it will relocate the bulk of itself there, leaving just over 1 kilobyte (1,200 bytes) of itself in conventional memory. If no expanded memory is available it occupies almost 12.5 kilobytes (12,784 bytes). The manual states inaccurately that 900 bytes and 8k are the respective TSR overheads; such discrepancies in a product are unfortunate.

Neither SDRES nor its *Windows* counterpart actually identify viruses, they simply display a warning panel suggesting that the user should run the scanner to identify and remove the virus. There is a slight extra delay whenever a new disk is inserted as the resident scanner leaps into action to read and scan its boot sector, but there is less of a time overhead when programs are executed.



The best feature about *Search And Destroy* is its ability to scan within such a wide range of compressed files.

Expanding Search Capabilities

Some scanners have the ability to detect that program files have been compressed using a dynamic compressor such as *PKLite* or *Diet* - one or two can actually scan within such compressed files (internal scanning), but most scanners only check to see whether the executable has been infected after it was compressed (external scanning). *Search and Destroy* not only scans dynamically compressed files both internally and externally but it also recognises all of the popular formats - *Diet*, *LZExe*, *PC Lite* and *PKLite*. It can also be configured to scan within compressed archive files such as those created with *ARJ*, *LHArc* and *PKZip*. Both of these options can individually turned on or off either as a command line option or from within the scanner program's 'options' menu.

I applaud *Fifth Generation* for providing this ability because so much commercial software is delivered in archives and often includes dynamically compressed executables. This recent trend can make life difficult if you diligently scan everything you receive (you do, don't you?). A word of warning though: *Search and Destroy* does not recognise all compressed formats, particularly those upon which much software is bought (for example the format *Lotus* uses for its *AmiPro Windows* program and *Microsoft's* compressed DOS 5 and *Windows* 3.1 distribution disks). So, while this is undoubtedly a very useful facility, its limitations must be realised. It is important to remember that there is nothing magical about distribution disks - viruses have been found on them.

Enabling the ability to scan within compressed files and archives does not seem to impact too heavily on the respectable scan speed. As a graphic example, I scanned a CD-ROM which contained a total of 6,683 files of which 546 are ZIP files which themselves contain a total of 6,137 files. With the ability to scan inside compressed files turned-off, this took 19 minutes 40 seconds or 5.2 files per second (by comparison, *Sweep* from *Sophos* took 15 minutes and 1 second in quick mode and *Dr Solomon's Toolkit* 13 minutes 6 seconds). However, enabling this function the scan rate dropped to 4.6 files a second.

Scanner Accuracy

In terms of its detection capability *Search and Destroy* is average - a less than ideal result for such a new product. It failed to detect two viruses known to be in the wild: Penza and Father, and also had some problems when it came to reporting a file that was infected with DIR-II. *Search and Destroy* identified the file as infected but displayed gibberish characters where the file name should have appeared.

Its detection of 356 out of 364 viruses in the now quite elderly *Virus Bulletin* 'Standard' test-set places it in the mid-range when it is compared to its competitors. It detected 735 out of 783 infected files in the enlarged test set which is also in the middle band. It does not, however, reliably detect viruses which employ the Mutation Engine, as it missed 90 infections out of a total of 1,536.

In naming viruses it fails to adhere to the widely adopted *VB* nomenclature used by much of the research community, which means that some of the names will not be immediately obvious.

The scanner has an option to display a listing of viruses it recognises. If, for example, you want to know whether it recognises one of the Spanish Telecom viruses, you need to know that it names the file-infection version Kampana but the Boot Sector versions are listed (confusingly) as both Spanish Telecom and Kampana.

Conclusions

For a brand new scanner *Search and Destroy's* results are a little disappointing. Not only did it not reliably detect the Mutation Engine, but it missed viruses known to be 'in the wild.'

In a market which is already saturated with software, any new product needs to have features which make it stand out. *Search and Destroy* does have the extremely useful ability to scan within compressed files and a free 24 hour help-line, but whether this is enough to recommend it given its rather mediocre detection results remains to be seen.

SEARCH AND DESTROY

Scanning Speed

Hard Disk:	
Turbo Mode (448 Kbytes/sec)	36 secs
Secure Mode (334 Kbytes/sec)	1 minute 29 secs
Floppy Disk:	
Turbo Mode	8 secs
Secure Mode	11 secs

Scanner Accuracy

'VB Standard' Test-set[1]	356/364	97.80%
'Expanded' Test-set[2]	735/783	93.87%
'In The Wild' Test-set[3]	125/128	97.65%
'MtE' Test-set[4]	1446/1536	94.14%

Technical Details

Product: *Search and Destroy*

Version: v1.0 (25.08)

Author: *BRM Technologies Limited*

Distributor: *Fifth Generation Systems*, Cliveden Office Village, Lancaster Road, High Wycombe, Buckinghamshire HP12 3YZ.

Telephone: (UK) +44 494 442223. (US) +1 504 291 7221.

Fax: +44 494 442225.

Price: £29.00 or $39.00 ($125 for monthly updates, $60 for quarterly.)

Test Hardware: All tests were conducted on an *Apricot Qi486* running at 25Mhz and equipped with 16MB RAM and 330MB hard drive. *Search and Destroy* was tested against the hard drive of this machine, containing 1,645 files (29,758,648 bytes) of which 421 were executable (16,153,402 bytes) and the average file size was 38,370 bytes. The floppy disk test was done on a disk containing 7 files of which 3 (25,508 bytes) were executable.

For details of the test-sets used please refer to:

[1] Standard test-set: Virus Bulletin - May 1992 (p.23).

[2] This unofficial test-set comprises 784 unique infections.

[3] 'In The Wild' test-set: Virus Bulletin - January 1993 (p.12).

[4] 'MtE' test-set: Virus Bulletin - January 1993 (p.12).

PRODUCT REVIEW 2

Keith Jackson

VIS Anti-Virus Utilities

VIS Anti-Virus Utilities (*VIS4*) claims to be a 'tightly integrated system which gives full and comprehensive detection of and protection from computer viruses'. *VIS* was last reviewed in the April 91 issue of *VB*. Anti-virus products have come a long way in the last two years: Has *VIS4* moved along as well?

VIS4 offers three main modes of operation - a *Windows* program, a full screen DOS application, and a command line driven DOS application. The software can be either keyboard or mouse driven, and a memory-resident utility is provided which can scan for virus signatures and/or verify file integrity dynamically.

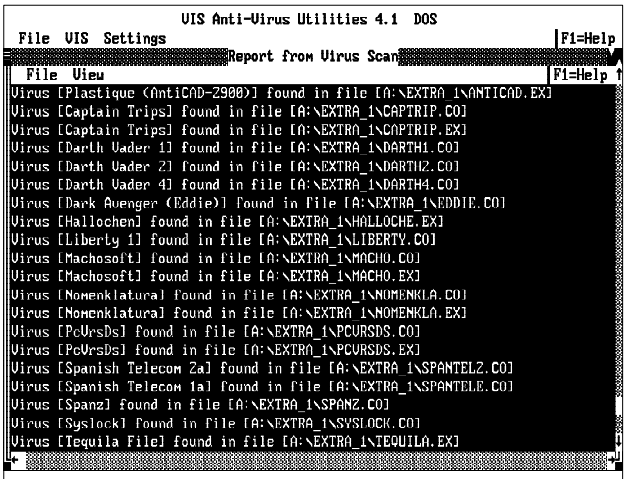
Documentation

The manual which accompanies *VIS4* comprises a 153 page book, which is well laid out and suitably indexed. It is easy to read, full of factual information, and contains detailed appendices which describe the various tricks used by viruses. I like the documentation - it is one of the better ones that I've seen, but I must query the decision to provide two manuals in one volume (one for DOS, and one for *Windows*), and then repeat many of the sections verbatim in both parts. This applies to whole sections (several pages), not just the odd phrase here and there.

Extra bumf which accompanies the manual comprises an errata sheet, a registration card, forms to obtain a copy of *VIS4* on 5.25 inch floppy disks and to order upgrades on a monthly or quarterly basis and, most interesting of all, a few copies of the *Computer Crime Unit's* Virus Report Form.

The latter inclusion I approve of most wholeheartedly. It is difficult for individual users to know what to do in legal terms if (god forbid) their computer ever does become infected by a virus. As an aside, the *Computer Crime Unit* at *New Scotland Yard* has done a good job in producing these Virus Report Forms. They are easy to understand, and explain how to report a virus infection to the police in simple terms, even down to describing how to save copies of the virus in a way that will stand up as evidence in court.

The on-line documentation is very extensive, ranging from a general explanation of how to use the various components of *VIS4*, descriptions of how viruses operate, what to do if an infection is found, and an extensive on-line encyclopaedia of the viruses known to *VIS4* (currently 1251 in total). The help system is easy to use (just click on highlighted items for further information), and unlike similar offerings provided with several *Windows* products, actually contains useful information.



VIS4 provides a highly accurate scanning engine, but at a price: it is one of the slower scanners available on the market.

Installation

VIS4 came on a single 3.5 inch, 1.44 Mbyte, floppy disk. Even though 5.25 inch floppy disks can be obtained (see above), I wonder where this leaves the owners of computers which can only read the 720 Kbyte version of 3.5 inch floppy disks. I have noticed that many companies have given up providing anything apart from 1.44 Mbyte floppy disks (*Central Point's PC Tools* had 5 of them!), and although this makes the vendor's life easier, it is definitely not progress as far as the smaller user is concerned.

Quite properly, the installation program advises the user to do a scan of the destination disk before installing *VIS4*. Installation can be performed from any source drive onto any destination drive (thereby avoiding many of the common installation program failings), and invites entry of a user message which is displayed whenever a virus is detected. This is very useful in large companies, and can be used to provide the user with a suitable contact telephone number for further help.

Although the installation process was itself straightforward, I did encounter some problems whilst trying to install *VIS4*. As usual I tried out various installation conditions (with and without *Stacker*, into a pre-existing subdirectory, etc), and I sometimes encountered the message 'Error in copying file' early on in the installation process. The available hard disk space was 6 Mbytes, so a lack of free disk space was

unlikely to be the problem. This error was then always followed by an 'Unable to find resource file' message. This problem needs investigation - it is most unlikely to be disk corruption as it occurred with several floppy disks.

When the above described problem occurred, then sometimes the installation program attempted to write back to the installation disk. This is unforgivable, especially during an error scenario, as anything could be happening. [*Total Control have had no other reports of any installation problems with VIS4 but are looking into this. Ed.*]

Scanning Speed

When *VIS4* scans a disk, several options are available which will scan for 'rare' viruses (defined in the manual as those viruses that are not widely circulated - not a very precise definition), and/or perform a memory scan before scanning of files commences. The drive(s) selected and the extensions of the files which are included can also be set as desired. All of these selections are made by either clicking on the appropriate response box with a mouse, or by using the keyboard. The output from the scan is written to a report file which explains what suspicious files have been found, and lists any compressed files. Browsing facilities are provided to inspect both this report and the one produced by the File Integrity tests (see below).

During a scan a continuously updated 'percentage complete' meter appears on the screen. For unknown reasons this went up steadily to 20%, then as the scan was complete it suddenly zoomed up to 100%. An inaccurate measure of how far the scanning process has got rather defeats the idea of providing such a feature in the first place. I am also rather by puzzled by the fact that if all drives are deselected, and a scan is then started, *VIS4* scans drive C: anyway. Surely this is wrong?

Using the default settings, a hard disk containing 11.2 Mbytes (in 247 files) was scanned in 1 minute 34 seconds. If rare viruses were excluded (1127 viruses out of a total of 1251), then the scan time reduced to 38 seconds, and exclusion of the memory scan further reduced this to 34 seconds. All scanning tests were carried out on a 33 MHz 486DX. For comparison purposes *Dr Solomon's Anti-Virus Toolkit* scanned the same disk in 13 seconds, and *Sweep* from *Sophos* performed the scan in 13 seconds in 'Quick' scan mode and 56 seconds in 'Complete' scan mode. The large decrease in scan time caused by the exclusion of rare viruses illustrates that *VIS4* appears to be suffering from the recent explosion in the number of PC viruses.

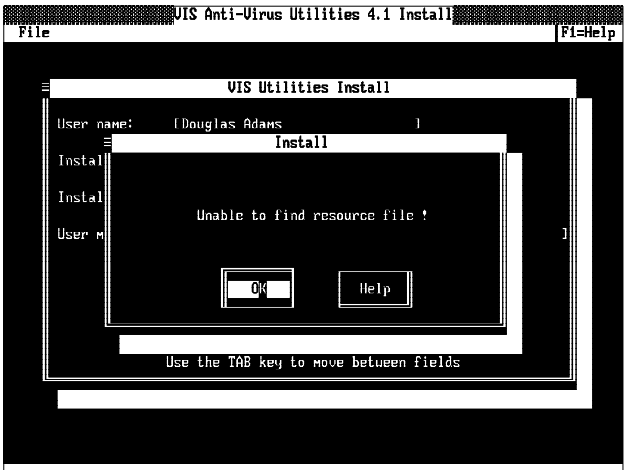
When *VB* first looked at this product (April 91) it was one of the fastest around: the review stated that the speed at which it can 'inspect a disk is very impressive'. This blistering speed advantage over other products has now largely gone.

The *Windows* front end provided with *VIS4* is very pretty, and is functionally identical to the DOS version. As somebody who has to learn how to use both versions, I appreciate this. The screen background provided with the *Windows* version of *VIS4* is particularly striking, showing a detailed drawing of some floppy disks, a first aid kit, and a magnifying glass. Very appropriate.

As ever, there is a 'downside' to the screen gloss provided by *Windows*: The scanning speed is significantly reduced. Using the same hard disk test described above for the DOS version of *VIS4*, scanning the entire disk took 2 minutes and 22 seconds (a 50% increase over the DOS scan time), reducing to 52 seconds (a 37% increase) when rare viruses were excluded, and reducing further to 39 seconds (only a 15% increase) when the memory scan was omitted. Note that a memory scan adds only 4 seconds under DOS, but it adds 13 seconds under *Windows*. Given that *Windows* manages the hardware all of this is unsurprising.

Scanning Accuracy

When last reviewed, this product correctly detected every single virus in the test set. This time around the test set is much enlarged, and *VIS4* detected all of the 215 virus samples bar one. Rather curiously the only virus that it failed to detect was Maltese Amoeba. This turned out to be because the file had been renamed after infection - *VIS4* uses the filename of Maltese Amoeba infected files to aid in identification. Even taking this slightly odd detection strategy (this virus is the only one detected in this way) into account, the scanning accuracy offered by *VIS4* on this test-set is excellent.



Don't Panic! *VIS4's* installation routine failed unpredictably leaving the user stuck at an 'Abort, Retry, Fail' prompt.

I commenced testing *VIS4* against 1024 Mutation Engine samples, but struck problems in this test. These Mutation Engine virus samples are stored on an old XT (4.77 MHz with a 8088 processor) that I keep solely for test purposes. In normal operation *VIS4* was very slow on this computer, but when it came across a possible Mutation Engine sample it slowed down to a measured rate of *36 seconds* per file.

On detecting a virus, *VIS4* displays a screen with various options and insists that the user touches a key or clicks the mouse. This cannot be disabled, or if it can then I cannot see how to do it. Given *VIS4*'s detection rate (see immediately below) this meant that I had to press a key after a random time duration of about 5 to 10 minutes. I struggled on manfully, and even resorted to pressing a key every time I walked past the computer, but I have to admit that I gave up after 373 Mutation Engine samples, of which *VIS4* had detected just 30 (8%). I leave it as an exercise for the reader to calculate how long this took, even without allowing for the time that *VIS4* sat there waiting for me to press a key.

File Integrity

VIS4 offers facilities to create and verify file checksums. It does this by maintaining a database containing information about all of the files on a disk.

According to the documentation, the database entry for each file contains the date/time stamp, file size, and a checksum calculated across the contents of the file. The on-line help states quite categorically that 'if any change, whatsoever, is made to the file then it will spot this change...'. This is claimed to operate 'if so much as one bit of the file has been altered'. Unfortunately it is not true. I used an executable COM file for testing purposes, and although *VIS4* always spotted a date/time alteration, it failed to detect alteration to the contents of this file unless these changes were made at the start of the file. However, working from the command line the integrity checker performed flawlessly. This turned out to be a problem with the documentation - a complete check can only be performed from the command line. The checksum algorithm is 'proprietary', and therefore it is impossible to comment on its security properties.

Memory-resident Utility

The memory resident utility provided with *VIS4* used to be available both as a device driver and as a terminate and stay resident (TSR) program. It is no longer available as a device driver. I was pleased to see the refreshing honesty in the accompanying notes explaining this recent adjustment, which admit that the change has been made because of 'several conflicts with third-party *Windows* software'. Most vendors simply deny there is a problem, I prefer the honest answer - it did not function properly, so we changed it.

However some problems still remain. I tested the scanning capability of the TSR utility by copying virus infected files from one drive to another. Not only did it recognise just 13 out of 29 files as infected, but after copying had finished the PC locked up so thoroughly that I could not enter any more commands. Further testing showed that experimenting with disabling various options made no difference to this. [*Total Control informs VB that it has since upgraded this part of its package. Ed.*]

Conclusions

VIS4 is refreshingly free from the nonsense that infects many of the anti-virus products. It does not make grandiose claims, and is honest to the point of admitting a false positive detection of the V2P6 virus on some *Harvard Graphics* files. Its claims to provide a 'tightly integrated system' are true, even down to the level that the DOS and the *Windows* version operate identically.

In its current incarnation *VIS4* does now exhibit operational problems. It *was* one of the fastest scanners around, that is no longer true. Its checksumming components do not work as the documentation describes them, and some component parts of *VIS4* exhibit problems. As for scanning detection, with the exception of the problems with the Mutation Engine, *VIS4* really is top notch.

I said 'Highly Recommended' at the end of my review of two years ago, but the problems described in detail above really do need sorting out before this product regains this distinction. Perhaps I should qualify this by stating that it is beginning to seem that extra technical manpower is needed to keep up with all the necessary changes.

Technical Details

Product: *VIS4*

Developer: Jim Bates

Vendor: *Total Control*, Unit 3, Station Yard, Hungerford, Berkshire RG17 0DY, UK, Tel: +44 (488) 685299, Fax: +44 (488) 683288.

Availability: For IBM or compatible PCs with a minimum of 512K of RAM, a hard disk with 1 Mbyte of free space, a floppy disk drive, and DOS 3.0 or later.

Version evaluated: 4.1G

Serial number: 40866

Price: £89 +VAT

Hardware used: (a) 33MHz 486 PC, with one 3.5 inch (1.44M) floppy disk drive, one 5.25 inch (1.2M) floppy disk drive, and a 120 Mbyte hard disk, running with MS-DOS v5.0, *Stacker* v2.01, and *Windows* 3.1 (b) 4.77MHz 8088, with one 3.5 inch (720K) floppy disk drive, two 5.25 inch (360K) floppy disk drives, and a 32 Mbyte hard card, running under MS-DOS v3.30

REVIEWS

Computer Security - Who's Solving the Problem?

This video is designed to increase awareness amongst staff at all levels about the potential security pitfalls inherent to computers. It follows the problems of a fictional multi-national organisation and demonstrates some of the day-to-day oversights, failures and general blunders which can cause inconvenience and even disaster.

The film itself is not Oscar-winning material, the storyline is not particularly gripping, the script is somewhat wooden and the acting fairly hammy. With more dramatic and tense direction the film might have achieved its objectives more successfully - to wit, to scare the living daylights out of everybody watching, not least senior management. Unfortunately, the effect on this reviewer was rather more soporific; perhaps the material was altogether too familiar.

The usual hoary old chestnuts are presented: a Post-it note with a password stuck to a PC glares at the viewer; a green caterpillar scuttles across the screen; the network has to be closed down setting everyone back a quantum leap; a password is cajoled out of an unsuspecting clerk; a discrepancy in the accounts reveals a £3 million fraud etc.

All this could make for highly entertaining viewing with a sharper script and tenser direction, but the overall effect seems remarkably flacid and dull. Even the commentary by various industry experts fails to enliven matters. There is a lot of emphasis on risk assessment and the need to involve top management in every aspect of security planning; unimpeachable advice this, but hardly earth-shatteringly original or revealing.

Overall, a rather uninspiring film which fails to emphasise the essential practical messages with anything like the vigour necessary to make sure these essential ingredients are properly chewed and digested. Who's solving the problem? The producers of this video are trying to but don't really succeed.

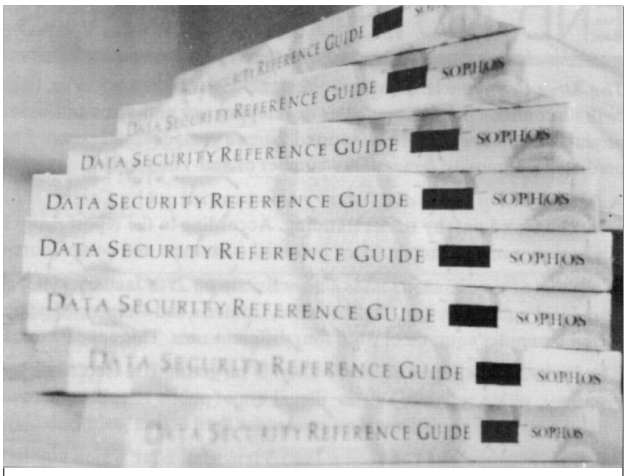
Computer Security - Who's Solving the Problem?

Producers: *Barclays PLC, Digital Equipment Company Ltd, The European Security Forum, Sophos Ltd, and Zergo Ltd.*

Format: VHS

Price: £49.95 + VAT + P&P

Available from: Positive Image Ltd, 148 Tooley Street, London SE1 2TU. Tel. 071 407 0265.



Data Security Reference Guide - still the only product catalogue with an ISBN number!

Data Security Reference Guide

Sophos Ltd has announced the release of the 1993/94 edition of the *Data Security Reference Guide*. The Guide, long since established as one of the standard reference books for those working in the industry, has been revised and now includes a section which discusses practical anti-virus measures which have been tried and tested in large organisations.

The *Data Security Reference Guide* features sections on all aspects of data security, including data encryption, secure erasure and authentication. Each section has an introduction which explains the relevant issues in that subject.

The much expanded section on computer viruses is well laid out and presents an accurate picture of the problem. A forty-seven page tutorial is provided on computer viruses. Starting from basics ('What is a computer virus') it rapidly proceeds with an in-depth guided tour of different virus types and techniques. The second part of the *Guide*, as before, is a catalogue of *Sophos* products. The release of the new book coincides with a major revamping of documentation and packaging of the whole *Sophos* product range.

Data Security Reference Guide - 1993/94

ISBN: 0-95-134203-7

Author: *Sophos Ltd.*

Price: £20.00 or $35.00

Available from: *Sophos Ltd*, 21 The Quadrant, Abingdon, OX14 3YS, England. Tel. 0235 559933.

END-NOTES AND NEWS

***The Xtree Company* is set to leave the anti-virus industry.** *Xtree*, the makers of *ViruSafe*, *ViruSafe/LAN* and *AllSafe* has announced that the company 'will discontinue publishing and/or developing any anti-virus and/or security products... All existing users who purchased anti-virus and/or security products from *Xtree* will be supported for one year, ending January 31st, 1994.' This move comes as no surprise to seasoned observers who have been predicted a slimming down of the number of anti-virus software manufacturers. Tel. +1 800 964 2490 ext. 3.

A *Department of Trade and Industry* report carried out by *Cooper & Lybrand Deloitte* says that the *1990 Computer Misuse Act* **is suffering from poor awareness and patchy understanding**. According to the report many firms believed that a prosecution under the Act would indicate a weakness in their business systems to shareholders, potential customers and competitors which could undermine confidence in them.

In a series of simultaneous raids across Britain on 27th January 1993, alleged members of the *Association of Really Cruel Viruses* were arrested. Officers from local units in Devon, Staffordshire, Cumbria and Greater Manchester, operating in conjunction with *New Scotland Yard's Computer Crime Unit*, seized computer equipment from four different sites. This appears to be part of a ruthless crackdown on virus authors in the UK.

The *Federation Against Software Theft* is targeting UK electronic bulletin board systems in a bid to stop the spread of computer pornography, virus programs, pirate software and the illegal use of public telephone networks. 'Pirate bulletin board operators are a significant problem in the UK and despite what the operators say there is no excuse for their action' said Bob Hay, *FAST* chairman.

S&S International **has released a 1993 Virus Calendar** which highlights 'virus free days predicted for 1993.' For a free copy of the calendar or useful information contact Jo Wheeler. Tel. 0442 877877.

Total Control has announced the launch of a new service, the *Virus Information Service Bulletin Board*. The board aims to provide details about viruses as well as free cure programs for some of the more common viruses. Tel. 0488 685299 or call the BBS directly on 0488 681291.

A new virus is reported to be in the wild in Germany. The sample arrived just as *VB* was going to press so it has not yet been fully analysed. The virus is 4000 bytes long, polymorphic and uses stealth techniques. It has also been reported specifically to evade *Central Point Anti-Virus*, which is believed to be included as part of *MS-DOS* Version 6.

A remarkable piece of anti-virus advice has been passed on to users by *International Data Security*. In a flyer advertising *McAfee Utilities*, users are told that if the checksum of any file they ship differs from those stated 'it may have been damaged or have options stored in it with the /SAVE switch. Run the program with only the /SAVE option to remove any stored options and then re-run VALIDATE.' Go to the bottom of the class...

Virus Bulletin has obtained an English translation of **the highly controversial *Der Spiegel* article** on the anti-virus industry. For a free copy of the article contact Victoria Lammer. Tel. 0235 555139.

VIRUS BULLETIN

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 Science Park, Abingdon, OX14 3YS, England

Tel (0235) 555139, International Tel (+44) 235 555139
Fax (0235) 559935, International Fax (+44) 235 559935

US subscriptions only:

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

Tel 203 431 8720, Fax 203 431 8165