April 1993 ISSN 0956-9979

VIRUS BULLETIN

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

CONTENTS

EDITORIAL

Law and Disorder

The 'hacking trial of the decade' has just finished at London's *Southwark Crown Court.* While such hyperbole is not uncommon in the popular press, it is certainly true that last week's trial of Paul Bedworth was a very important test of the 1990 Computer Misuse Act.

Unfortunately, rather than being a milestone in the fight against computer crime, it may well become more of a millstone around the necks of those who are in the unenviable position of trying to enforce the law. Despite having admitted hacking into many different computer systems, the verdict was a disaster for the prosecution: Paul Bedworth was found not guilty.

The attentive reader will have noticed that if Bedworth admitted to hacking computer systems, his charges cannot have been simple unauthorised access and modification of computer software. Indeed, if such charges had been made, it is most unlikely that Bedworth would have had any choice but to plead guilty. Such a case would probably not be conducted at a Crown Court, but at a Magistrates Court, and the maximum sentence would have been six months imprisonment or a £2000 fine. Hardly commensurate with tens of thousands of pounds of damage.

While nobody except the twelve members of the jury knows why they returned a 'not guilty' verdict, it is probable that the decision was taken because they felt that Bedworth was 'compelled' to hack. It is also possible that there was an element of 'helping the underdog' - after all, how could a nineteen-year-old man cause such universal pandemonium? Bedworth stood like David before the mighty corporate Goliath - and as in that biblical fight, it is hard not to think of Goliath as anything other than a bully.

New Scotland Yard's Computer Crime Unit produced vast amounts of evidence showing precisely what Bedworth's actions were. It is believed that one of the reasons for which the conspiracy charge was made was that the whole group of hackers could be tried together. The sheer bulk of evidence made separate trials almost impossible. 'If we had gone for individual hacking charges, the indictment would have been the size of a telephone directory', commented Det Sgt Barry Donovan, an investigator on the Bedworth case.

Part of the problem seems to be that computer crime is not perceived to be particularly anti-social. Many foolishly blame the IT Managers of the systems which have been hacked into. Such twaddle does little for the image of the industry - and even less for those who fight computer crime. Sadly, this attitude exists not only among members of the computer underworld, but among 'consultants' and 'advisors'. Nobody would doubt that leaving default system passwords installed on a mainframe is like leaving the car keys in the ignition. However, the idea that it is a system manager's fault if the system is hacked is ludicrous. If a house is broken into, no matter how thin the glass on the windows, there is no doubt whether a crime has been committed. Why should a computer system be any different?

The defence that was offered claimed that Bedworth's obsession with computers prevented him forming the intents that were necessary for the charges. Anybody who has ever met a computer hacker will tell you that they are all, to a greater or lesser extent, obsessed with computers. Generally loners in the real world, they commonly have a hacking alter ego which is often unrecognisable to those who know them.

The original definition of a hacker was someone who could find elegant solutions to computer problems. Now however, the connotations are such that one immediately pictures a 'computer junkie' hunched over a keyboard in his bedroom. The crumpled Led Zeppelin T-shirt and blue jeans are optional - but the image is hard to lose.

Does this spell the end for the Computer Misuse Act? Do hackers now have *carte blanche* to wreak havoc across computer networks everywhere? Contrary to much editorial opinion, the Computer Misuse Act does not lie upon the floor in tatters, although the events of the last month have not helped it one iota. Hackers will still be pursued by the appropriate authorities 'with vigour' according to DC Noel Bonczoszek of the *Computer Crime Unit.*

It is important to remember that the result of the Bedworth case does not set a legal precedent - anyone who tries a similar defence will have to convince the jury that they were not in control when they hacked into particular systems. Indeed, the picture portrayed by the Bedworth defence was of a young man for whom hacking was a compulsion, someone who would become angry if anyone came between him and his computer. However Bedworth became a hacker *before* the 1990 Act, and could therefore argue that he was addicted to hacking before it became illegal. It should prove more difficult to use the same defence for those who became hackers *after* the 1990 Act.

It seems unlikely that *anybody* who relies on mainframe systems will be happy with this decision - and the silver lining to this whole sordid affair is that it may well prove to be the catalyst needed to beef up the laws within the UK. Rather than waiting for the next Paul Bedworth to emerge, now is the time to strengthen the 1990 Computer Misuse Act so that the maximum punishment in some way reflects the amount of damage which can be caused.

NEWS

Phrack is Back!

Phrack, the underground 'hacking and phreaking' magazine, is set to return. However, *Phrack* will no longer be an electronically distributed periodical: in its new incarnation, it will be distributed as a legitimate news-stand publication, with its own ISSN identification number.

Subscriptions to *Phrack* will cost $100 for four quarterly issues. The new editor, Chris Goggans, insists that *Phrack* will still be free to 'the people' but that corporates will have to pay for it. Goggans claims that 'the new *Phrack* will be so good that subscribers will find it worth the annual charge'.

With *Phrack* now joining the ranks of *2600* and *AVDQ*, it seems likely that the many corporate users will have to suffer the ignominy of having to register and pay to receive it. Goggans can clearly see the funny side of this: 'It's kind of ironic', he laughed❑

What To Do When The Magic Fades Away...

A new strain of the CMOS1 virus has been found. The virus, named EXEBUG2 is successful in its attempts to disable a clean boot on some infected machines. The most sacred of magic objects seems to have been reduced in status.

The virus is completely capable of 'faking' a clean boot on some machines (such as those equipped with a certain version of the AMI BIOS). Like the CMOS1 virus, EXEBUG2 operates by altering the information stored in the CMOS memory so that the computer always boots from the fixed disk drive.

Fortunately there is no need for global panic, as the virus only operates correctly on a small percentage of machines. However, it does highlight the danger of having a simple and easy method of selecting the boot device from within software - BIOS manufacturers would do well to take note of this when designing new systems❑

How (not) to Organize a Conference

The Technical Editor of Virus Bulletin attended a computer security conference in New York in March - an annual event at which virus researchers gather, give talks and share ideas with one another. He reported as follows:

Last year, the organisation was bad - this year it was a disaster! I discovered that I was supposed to chair one session, but I only knew who the speakers were an hour before. The first speaker's talk was about something different to what the schedule indicated, the second speaker just managed to show up in time (it seems that nobody informed him that his paper had been accepted until the day before), and the third speaker had already left. Probably nobody had told him that he was supposed to speak!

There were numerous minor hiccups: the proceedings never appeared, some people (including myself) never got their name badges and no coffee was provided during the coffee breaks. However, there were also more serious problems - some well-known members of the anti-virus community (including one whose picture was on the cover of the conference brochure) got thrown out by the security staff, and (worst of all) the food was either missing or in short supply... I wistfully thought of the *VB* conference gala dinner as I watched somebody else grab the last sandwich.

Only one humorous event dispelled the gloom: one of the speakers was an 'anonymous' member of the Phalcon/Skism group - or so he thought. To his surprise he was wrong... as a huge sign saying 'Welcome, Simon B.....' greeted him as he entered the conference hall. I wonder who could have been responsible for that? [*Me too. Ed.*]

Some of the speakers at this conference have already promised to boycott next year's event, while others have demanded that ACM and IEEE withdraw their support. However, what will actually happen remains to be seen❑

Virus Prevalence Table - February 1993

Viruses reported to *VB* during January 1993.

Virus	Incidents	(%) Reports
Form	15	29.3%
Spanish Telecom	7	15.9%
New Zealand 2	5	11.4%
Eddie 2	3	6.8%
Cascade	2	4.5%
Joshi	2	4.5%
NoInt	2	4.5%
Tequila	2	4.5%
V-Sign	2	4.5%
DIR-II	1	2.3%
Disk Killer	1	2.3%
Flip	1	2.3%
Power Pump	1	2.3%
Total	44	100.0%

IBM PC VIRUSES (UPDATE)

Updates and amendments to the *Virus Bulletin Table of Known IBM PC Viruses* as of 25th March 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

ARCV-9 - CN: There are actually two viruses called ARCV-9, 745 and 771 bytes long. To confuse things further, the original sample of this virus was infected with both variants. The viruses use variable encryption, similar to that used by the PS-MPC-generated viruses, which the author seems to have studied carefully.

ARCV.Joanna - CN: A 986 byte English virus, which contains the text 'This is Dedicated To the Girl I Love', as well as a girl's name. The virus is slightly polymorphic, but can be detected with the following pattern, which should be used with care because of the large number of wildcards.

```
ARCV.Joanna     BE?? ??B9 E201 BF?? ??FC AD?? ???? ABE2 F9
```

ARCV.Sandwich - CN: This 1172 byte virus is written by the same author as Joanna, and closely related.

```
ARCV.Sandwich   BE?? ??B9 3F02 BF?? ??FC AD?? ???? ABE2 F9
```

Armagedon.1074 - CR: Very similar to the original variant, but 1074 bytes long. Detected with the Armagedon pattern.

Backfont.896 - ER: The Backfont family includes the viruses originally reported as '905' and '765'. This new, 896 byte variant has not been fully analysed.

```
Backfont.896    760A 2680 7C01 3A75 0626 8A14 80EA 40B4 36E8 DEFF 3DFF FF74
```

Barrotes - CER: A 1310 byte virus from Spain. This 1310 bytes virus activates on Jan 5th, displaying the message 'Virus BARROTES por OSoft', and corrupting the boot sector.

```
Barrotes        3D00 4B74 03E9 DB02 5053 5152 1E06 5657 2E89 164E 012E 8C1E
```

Beer.3164 - CN: The third variant of the Beer virus, probably by the same author as the other two.

```
Beer.3164       FA90 80FC 3B75 03E9 31FF 3D00 3D74 0F3D 023D 740A 80FC 5674
```

Burger.560.Liquid - CN: Yet another variant of this old overwriting virus. Detected with the Burger pattern.

Danish Tiny.177 - CN: Detected with the Danish Tiny.Brenda pattern. Does nothing but replicate.

Danish Tiny.180 - CN: This virus does nothing but replicate.

```
Danish Tiny.180 8BFA B903 00CD 2180 3DE9 7407 B44F EBDC EB69 90B8 0057 CD21
```

Dark Avenger.1800.Quest - CER: Almost identical to the 1800 byte variant, but the text messages have been changed. Detected with the Dark Avenger pattern.

Emmie.2620 - CR: A new variant of this stealth virus, which is probably of Israeli origin. This variant is 2620 bytes long, and is detected with the Emmie pattern.

Gotcha.F - CER: 732 bytes long and similar to the other known variants.

```
Gotcha-F        8BF0 BF00 01B9 1800 F3A4 0EB8 0001 50B8 DADA CD21 80FC A574
```

HH&H.4093 - CR: Very similar to the original variant, but two bytes longer.

```
HH&H.4093       50B9 FD0F 8B1E 0101 81C3 1501 8037 ??43 E2FA
```

Intruder.1326,1440,1988,2336 - EN: Four variants of length 1326, 1440, 1988 and 2336 bytes, which are very similar to previously known variants. Detected with the Intruder pattern.

Itti.Toxic - CN: A 171 byte non-remarkable overwriting virus.

```
Itti.Toxic      998B CAB8 0042 CD21 B440 B9AB 00BA 0001 CD21 BE95 00AC 50AD
```

Jos - CR: This 1000 byte virus originated in Romania as the alias 'Romanian Jabberwocky' indicates.

```
Jos             3C03 74F7 3C00 7456 3CFF 7504 BFAA 55CF 3CFE 75E7 B451 CD21
```

Leprosy.FVHS.1644 - CN: A 1644 byte overwriting virus, which is detected with the Leprosy.Silver Dollar pattern.

Lovechild.2710 - CR: This is a new version of the Lovechild virus, and just like the original it might be able to infect the MBR as well.

```
Lovechild.2710  33C0 8EC0 E800 005E 8BEE BFE0 01E8 7801 FC26 813D 7634 7503
```

Loz-693 - CR: A 693 byte virus. Awaiting analysis.

```
Loz-693         5706 521E 2E2A 26B4 0374 03E9 DF00 FC8B F2AC 0AC0 75FB 0E07
```

Matura.632 - CN: A 632 byte virus. Awaiting analysis.

```
Matura.632      83E1 1F83 F91E 74DE 3E8A BE09 0380 E701 80FF 0175 0EB4 43B0
```

Milan.BillMe - CN: This is an overwriting virus, and therefore very unlikely to spread. As it showed some similarity to the Demon virus as well as to the other Milan viruses, the relationship of these viruses has now been re-examined, and as a result the Demon virus is now classified as a member of the Milan family.

```
BillMe          02EB 02EB EFB4 2ACD 213C 0274 0BB4 09BA DE01 CD21 B44C CD21
```

Npox.609 - CR: A 609 byte encryted virus. Awaiting analysis.

```
Npox.609        5D55 3E8A 865D 02B9 3A02 2E30 4600 45E2 F9C3
```

PCBB.1141 - CER: This virus is related to the 1129 byte variant originally reported as 'Plaice', and is detected by the same pattern.

Phalcon.Elvis - CN: Very similar to the Ministry variant, but 1250 bytes long and contains a different text message.

```
Elvis           BE15 0103 3606 018A 24B9 A704 81C6 2E00 8BFE AC32 C4AA E2FA
```

Pixel.Cheef - CN: This 300 byte variant activates the third day of any month, overwrites a part of the disk and displays the message 'Happy Birthday,Cheef!'. It also contains the string 'Moscow', which may indicate where it originated from. The virus is detected with the Pixel.277 pattern.

Pixel.762 - CN: Detected with the Pixel.936 pattern. This virus contains the text 'LiquidCode', but that string can also be found in several other recent viruses, belonging to different families. Presumably these viruses are created by the same person, who does not seem to have the technical knowledge to write a virus from scratch.

Polish Tiny.176 - CN: A small, 176 byte virus which does nothing of interest.

```
Polish Tiny.176 00B1 04CD 21B8 0242 33C9 33D2 CD21 B440 8D94 FDFF B1B0 CD21
```

Problem.854 - CER: Almost identical to the 856 byte variant reported earlier.

```
Problem.854     509E 8BE5 8946 0658 E803 005D 9DCF 2E8C 1664 032E 8926 6203
```

PS-MPC.Alien - CR: This 392 byte virus is not encrypted, which is quite unusual for a virus in this family.

```
Alien           595A 1FE8 3300 EB89 B903 0051 2BC1 BE8C 01BF 6B00 A5A4 C644
```

PS-MPC.Gold - CER: Encrypted, 612 bytes and detected like other encrypted PS-MPC viruses. Other new variants in this family are WaReZ (1803 bytes, CEN), Jo.942 (942 bytes, ER), Cinco (885 bytes, CR), Tim.401 (401 bytes, CN), Jo.916 (916 bytes, ER) Tim.301 (301 bytes, CN) and Tim.515 (515 bytes, EN).

PS-MPC.Grunt.203 - CN: A 203 byte virus awaiting analysis. Several viruses called Grunt have been made available, but it is not yet clear whether they should be considered to belong to the same family or not.

```
Grunt.203       B800 3D8D 96EE 01CD 2193 B43F 8D96 CA01 B903 00CD 218B 86EA
```

Screen - ER: A 1014 byte virus. Awaiting analysis.

```
Screen          9C50 518B C880 FD4B 7413 2EA0 EA00 3C00 7547 80FD 2C74 0F59
```

Semtex.1000.C, Semtex.619 - CR: Two new variants of this virus, previously called CSFR-1000.

```
Semtex          803E 0000 5A74 1740 8EC0 2681 3E00 00CD 2075 0526 293E 0200
```

SillyCR.178 - CR: A simple 178 byte virus that does nothing but replicate. The virus may be related to the AT family, but requires further analysis.

```
SillyCR-178     9C3D 004B 7558 B802 3DCD 2172 5193 0E1F BA3E 00B9 0300 B43F
```

StinkFoot.2E - CN: This new 1254 byte variant is very similar to some of the other known variants, and is detected with the StinkFoot pattern.

SVC.1228 - CER: This virus appears to be an old member of the SVC family, although it was discovered recently. It is detected with the SVC 3.1 pattern.

Timid.320, 371 - CN: This 320 and 371 byte variant are detected with the Timid.306 and Timid.305 patterns respectively, but have not been analysed fully.

Trivial.64 - CN: This 64 byte, overwriting virus contains the word 'Trident', possibly indicating it is written by the same person who wrote the TPE.

```
Trivial 64      B802 3DBA 9E00 CD21 B740 BA00 0193 88E1 CD21 B43E CD21 B44F
```

Trivial.81,30.D - CN: Two simple, destructive viruses.

```
Trivial.81      B802 3DBA 9E00 CD21 C536 4D01 9392 B151 B43F CD21 5133 C9B8
Trivial.30.D    BA9E 00CD 21B7 4093 BA00 01B1 1ECD 21C3 2A2E 2A00
```

Trivial.PopooLar - CN: A simple, 145 byte overwriting virus.

```
PopooLar        B41A 8D56 80CD 21B4 4EB9 2700 5ACD 2172 07E8 0D00 B44F EBF5
```

Two Minutes - CN: A 441 bytes virus written by a member of the (now defunct) ARCV group. Awaiting full analysis.

```
Two Minutes     B42A CD21 C784 E602 4F4D C684 E302 2AC6 84E8 0200 80FA 1F75
```

V-163 - CR: This is one of the rare viruses to be actually encountered 'in the wild'. It is 163 bytes long, and does nothing but replicate.

```
V-163           80FC 4B75 4050 5352 511E B892 3DCD 218B D8E8 3B00 B43F B104
```

VCL.Viral Messiah - CN: Due to an oversight, one of the sample viruses included with the VCL toolkit had not been covered before. This is a 702 byte overwriting virus. The following signature should be used with care, due to the high number of wildcards.

```
Viral Messiah   0701 B954 0181 ???? ???? ??E2 F8C3
```

Vienna.604, Vienna.618.B - CN: Two minor variants, both detected with the Vienna-4 pattern.

Vienna.700 - CN: A non-remarkable variant by the RABID group, which has been modified by inserting 'do-nothing' instructions into the code. This variant is detected with the Vienna(1) pattern.

Vienna.851 - CN: A 'buggy' variant, not particularly interesting. Detected with the Vienna (1) pattern.

Vienna.MD-499 - CN: A 499 byte virus. Awaiting analysis. Two closely related variants, 498 and 577 bytes in length, also exist, and are detected with the same pattern. The third variant in this group is 354 bytes long.

```
MD-499          241F 3C1F 74EF 83BC 8500 0075 E881 BC83 0000 FA77 E081 BC83
MD-354          B903 008B D683 C20D CD21 72B6 3D03 0075 B1B8 0242 B900 00BA
```

Vienna.New Years - CN: A 697 byte virus. Awaiting analysis.

```
New Years       ACB9 0080 F2AE B904 00AC AE75 EDE2 FA5E 0789 BC30 008B FE81
```

Vienna.Vio-lite - CN: A 988 byte variant containing the text 'Vio-Lite, TAA, Virulent Graffiti, (k) 1992'.

```
Vio-lite        ACB9 0080 F2AE B904 00AC AE75 EBE2 FA5E 0789 BC32 018B FE81
```

Vienna.Violator.Baby - CN: A 1000 byte variant, very similar to the Violator.Arf variants, but contains the text 'by by, baby'. It seems to have been created with a tool, which allows the user to specify the activation date and the text message to display. Detected with the Violator pattern.

X-1.570 - EN: New variant of the X-1 virus, 570 bytes long.

```
X-1.570         0E1F 8C06 7403 8C16 7603 8926 7803 8CC8 0510 0033 DB4B 8BE3
```

Yeke.1204 - ER: An Italian (?) virus, which is awaiting analysis. One variant, 1076 bytes long is also known.

```
Yeke.1204       BF1B 000E B976 04E8 0000 5D81 ED9F 041F 03FD FC07 8BF7 AC04
Yeke.1076       BF1F 000E B9F2 03E8 0000 5D81 ED1F 041F 03FD FC07 8BF7 AC04
```

Zherkov.1940 - CER: Very similar to the 1915 byte variant.

```
Zherkov.1940    5006 1EE8 0000 5E2E 8A44 E73C 0074 118B FE83 C71A 90B9 DC06
```

DISCUSSION

James Beckett

In a Squeeze

Anti-virus software documentation advises checking newly-received software for known viruses before you let it anywhere near your system.So let's say you scan a new disk and it comes back 'All OK'. Content that you have checked for viruses, you type A:INSTALL... Unfortunately, the only file that has actually been checked is 'PKUNZIP.EXE': the INSTALL.BAT file runs PKUNZIP to copy 2 Mbytes of software to your hard disk - including a virus-infected executable. The result would be an infected hard drive.

Incidents like this are all too common - so why don't most anti-virus packages scan inside compressed software?

Slowing Dowwwnnnn.n.n...

One of the problems is that compressed files may sport any of several name extensions; rather than just the canonical 'executable code' files *.COM, *.EXE and *.SYS (etc to taste), you must examine ZIP, LZH, ARC, BOO, PAK, LHA and more. Furthermore, these are just the default names; any extension is valid. Accessing all the files on the disk will make a scanner very slow - and this still leaves the problem of scanning compressed files within compressed files!

Apart from sheer weight of numbers, there are technical concerns in scanning compressed files: one has to cater for a multitude of different compression techniques, including unpublished ones written in-house.

Any unrecognised decompression system will be missed by the scanner, resulting in a 'clean' statement when in fact it cannot be sure, even when the virus within is one which it *does* know about. While not quite as tough as keeping up with new viruses, keeping up-to-date with new compression systems is a substantial task.

As an additional bonus, most compression programs are protected by copyright - to incorporate a routine will require either a developer's toolkit from the writers, with the inevitable vast licensing fees, or the reverse-engineering of their code, which invites prosecution.

Horses for Courses

Advice frequently given is that a user should simply uncompress the files onto the hard disk and re-scan them there. This seems adequate for small operations, but in a large organisation, virus checking and import of disks is often automated for the convenience of users. This makes this suggestion rather unwieldy.

Even more of a problem is posed by programs which uncompress themselves in memory and continue to run - here, a virus could be attached to the program within the compression, without any way of writing the decompressed file out to disk to facilitate checking.

It is often argued that for a program to be infected *within* the compression, it must have been infected at source. However, this does not have to be the case: it is not unknown for a virus writer to uncompress a package, infect the executable and re-compress it. Either way, users rightly expect their favourite scanner to detect viruses 'hidden' in this way.

Do we Really Need it?

If scanning for compressed files is going to slow down our system it is worth considering how much users really need this facility. A well-run organisation should not allow unauthorised software to be used anyway - games and utilities trawled from bulletin boards or passed between friends have always presented the highest risk and should be prohibited on 'work' computers.

This means that there is usually little real need for users to be passing compressed files around the office - only as disks enter the system do they need to be checked.Therefore, a scanner only needs to scan self-expanding files and the two or three really widespread compression formats. Everything else can be decompressed and scanned when it is imported.

Conclusions

This sounds as if the only route to security is a draconian regime of procedures and regulations, and a loss of some degree of individual freedom. Users may balk at this, but it is they who are going to scream loudest when their data disappears.

The home user has different problems to the corporate user; much of his software may in fact come from bulletin boards (many useful utilities are freeware or shareware) and is most at risk. However, it is reasonable to expect him to scan his software as and when he decompresses it.

For the corporate user, some level of compressed file scanning is required. All dynamically compresses executables should be examined, and the most popular compression techniques should also be covered for in-house use. Whether this represents a reasonable target for the anti-virus software vendors remains to be seen, but the bottom line for them is selling their product, and if they don't provide what users want, they won't earn their next meal.

FEATURE

Winn Schwartau
Inter. Pact

Using Security Modelling to Combat Viruses

As a security architect, I hold the view that a virus infection is a violation of good security practices. Along with other types of computer and network security vulnerability, viruses are but one aspect that needs to be considered. In fact, I strongly believe that a virus is no more than a remote control hacker with particular goals and techniques unique to the spread of malicious software.

In the age of downsizing, shrinking budgets and a general corporate reluctance to invest in computer life-insurance, IT managers have to make decisions on where to spend their sparse dollars, pounds, marks or francs. Thus in many instances, organisations have had to decide between installing computer security mechanisms or anti-virus software. This fiscal conundrum has been based upon a false premise, in part created by the media and fuelled by many anti-virus vendors in the hope of increasing their sales.

In many ways, it can be argued that the success of the 'Virus Busters' has been at the expense of good security practices in most organisations. Because of widespread naïvety on the part of the user community, a false sense of security is achieved when anti-virus software is installed. The justification that no additional monies need be spent on security and that the cost of the anti-virus software is sufficient expenditure, will be found to be penny-wise and pound-foolish.

The point is clear: anti-virus software is in no way a replacement for security, whereas good security packages provide excellent mechanisms to protect systems against viruses.

Security Mechanisms

Most professional security practitioners adhere to seven basic criteria by which the functionality of a computer security package is measured. In Europe, the *ITSEC* (*Information Technology Security Evaluation Criteria*) utilise these functional criteria and provide a measurement of the product's assurance - that is, whether the product performs as it claims to do.

In the United States, the *Orange Book* (D2, C2, B1 etc.) is more formally known as *TCSEC*, or *Trusted Computer Security Evaluation Criteria* and is being superceded by more comprehensive criteria. The latest commercial draft revision is in the comment stage as I write.

These criteria offer an extensive toolkit by which the security manager of an organisation can simultaneously achieve:

- A specified and desired level of security
- Protection against viruses.

As I describe these mechanisms and how these two goals can be reached using the same piece of software, keep in mind the fact that these techniques fight viruses in a *proactive* manner rather than a defensive *reactive* manner, which has been one long-standing criticism of the anti-virus field. A well implemented security scheme will protect against unknown viruses with no need for updates.

The conceptual core of any security system is called a Reference Monitor. The Reference Monitor is, in effect, a 'traffic cop', which watches what happens to the computer at all times. The premise is simple. Every request made of the system, either by the user or by a process, must be mediated by the Reference Monitor for approval.

The Reference Monitor only offers two choices to a system request: Go or No-Go. Either the event may occur as requested or the process is halted in mid-stream. The rules by which the Reference Monitor operates are based upon how these seven criteria are implemented. The proper use of these rules will keep your system protected from viruses, and perhaps you will be lucky enough to prosecute the distributors of the virus armed with hard evidence.

Criterion 1: Identification and Authentication

A computer, a network or indeed any system where information of value is stored should have a 'lock' on the 'front door'. Anybody who gains access to a computer should have to identify themselves to the system, and the user should have the means to prove his legitimacy.

Many popular methods can be used, ranging from simple passwords and ID codes to stringent biometric techniques. Pick one, and use it religiously. As you will see, this is critical, not only for keeping viruses off the computer, but in identifying and apprehending offenders and tracing the source of offending code.

Many security tools offer high degrees of flexibility at the 'front door'. Defining which days of the week or which hours of the day the user may access the system will help maintain system integrity.

Criterion 2: Object Re-use

This somewhat arcane term is designed to keep one user from passing himself off as another, or discovering private information about other users. There should be no transfer of

compartmentalised information from one user to another without specific authorisation. When a user leaves the system, all traces of his activity should be erased from RAM, all registers should be reset, and the contents of all temporary files securely erased from the system.

It should be obvious at this point that the purpose of criteria 1 and 2 is that the system knows who is using it and when.

Criterion 3: Audit Controls

A good security system offers a secure storage area which records computer transactions and requests. The audit log should record both successful*and* unsuccessful attempts to access the system. A log of the programs executed, and the results of the Reference Monitor's mediation should be saved for later analysis, especially if a virus or other security breach is detected.

Many criteria add the category of accountability, but here the aim is to ensure that the audited events can be attributed to a specific user. The audit log can be used to demonstrate that a particular user is attempting to bypass security measures which may call for some action to be taken on the part of the organisation. In the extreme, the audit logs may provide strong legal evidence in a criminal prosecution.

"anti-virus software is in no way a replacement for security"

Criterion 4: Access Control

Once the user has been identified by the system by Criterion 1, and isolated by Criterion 2, the Reference Monitor rules are put in place for that particular user. These access control rules govern factors such as what the user is authorised to access and modify, or which resources he has access to. This one criterion is an incredibly strong prophylactic against viral infection of the computer.

The user may, for example, be able to use all files and programs on Server 1 and his own hard disk, but no others. In fact, he should not even be permitted to*see* those resources to which he has no legitimate access. Alternatively, the user may be able to read certain files but make no changes to them, or files will be marked as being 'execute only', whereby the user cannot read or modify the program, but merely utilise it. Access control mechanisms can also be applied to floppy disks, and may govern the kind of files, if any, that can be copied or executed to or from them.

The access control rules which govern the extent of the Reference Monitor not only provide excellent viral insulation, but are also a vital part of any good security system.

This is a good place to consider a couple of examples. All system files should be automatically protected from modification and erasure. COMMAND.COM is a common target, but viruses can attack system files that most users are unaware even exist on the computer. CONFIG.SYS, AUTOEXEC.BAT, the Boot Sector and FAT tables should be inaccessible by anyone except the system administrator.

All EXE, COM, BAT, SYS and OV? programs must be similarly protected. The best method is to set them all to Execute Only. The executables cannot be copied, read, debugged or otherwise manipulated. In a system with proper security mechanisms in place, if a virus attempts to copy itself to an executable, two things will happen:

- The system will stop and the user will be alerted that an unauthorised write attempt has occurred.
- The audit trail will record the attempt providing the offending source file which contains the virus.

Note here that the kind of virus is immaterial. The common Cascade virus or an unknown 'Mutating-Super-virus' will be stopped dead in its tracks. This is called 'active containment'. In contrast to signature string databases which are unable to protect against unknown viruses, viral activity is automatically halted by a properly ruled Reference Monitor.

One argument against this approach is that some (thankfully few) programs write back to themselves. [*For example SETVER in MSDOS. Ed.*] This does not have to be a problem. Establish a security domain, possibly a subdirectory, where the access control rules permit only those executables within that domain to write to themselves. They cannot write to other programs, and other programs cannot write to them. An entire file server could be given such attributes depending upon the size of the organisation.

Another way to control access is to limit the use of floppy disks. They can be totally disabled within certain departments or throughout the entire organisation. This may be too extreme within most companies, so it may be preferable to set the access control rules to something like

```
A:*.EXE   No Access
```

This means that executables may not be copied to the hard disk from the floppy, may not be copied to the floppy, nor may they be executed from the floppy. A corollary benefit to the obvious virus protection that will appeal to attorneys is that the site licences are maintained, obviating many potential legal liabilities in the area of copyright. It also

prevents unwanted and unproductive games being used on company machines. Any attempts to violate this policy are recorded in the audit trials.

For the majority of users, floppy disks then become transporters of data *only*. I like that a lot, and so should most network administrators.

Users who understand the niceties of DOS or other operating systems could rename an EXE file to DAT, import it and rename it back. However, the access control rules can prevent this: by limiting access to DOS commands through the chosen access control rules.

How many DOS commands are really needed by the average user? That is precisely what the security policy will help to determine. The audit trail will record even command prompt requests and denials, which will tell the administrator who the closet hackers are.

The access control tables should offer total flexibility in assigning rights to files, directories, drives or servers. In security parlance, this is called granularity, a measurement of how finely we can tune the control over the system.

For those who feel any of these approaches are too limiting (which in a well managed networked office would be surprising), there are other answers which overcome any objections and still provide the same results.

Criterion 5: Integrity

Integrity is a familiar concept in these pages. It means a mechanism which detects if a file has been altered from some known, predetermined state. CRC or Cyclic Redundancy Checks are one inexpensive and simple way to enforce integrity. At the high end, X9.9 Message Authentication methodologies use strong cryptographic techniques (such as DES) and are employed to ensure the veracity of financial and high security records.

Any modification to a file of concern will be detected. For virus control purposes, recording changes to executables should suffice, but some organisations prefer to have additional controls. For example, files in a public directory may be given integrity controls. Thus the employee prankster cannot change a company-wide memo with impunity.

Criterion 6: Confidentiality

For our purposes, confidentiality means encryption of data. To the security practitioner, encryption prevents unauthorised people from reading files, or eavesdropping on data transmissions. For viral protection, encryption offers protection that anti-virus software simply cannot.

Consider an encryption model and its function in a typical workstation.

➤ All the files written to disk are automatically encrypted.

➤ All the files read from disk are automatically decrypted.

What does this mean? It allows the user to process data and run programs in their native plaintext mode, but all file storage is secure... *even if the whole computer is stolen!* This is particularly attractive on portables.

This argument can be extended to floppy disks.

Assume that automatic encryption is applied to floppy disks. Thus all files read from floppy disk are decrypted, and all files written to the floppy are encrypted. What would happen if an unecrypted text file were read into the machine? It would be decrypted, and decrypting a text file is effectively the same as encrypting it.

"Passive techniques have proven to be ineffective, antiquated, and cumbersome"

So, someone who tried to import executables as data on a floppy would get simply import gibberish, which would almost certainly crash the machine if it were executed.

Consider the implicationsif a company sets up its computers in this way. The benefits are enormous. Besides the virus protection, all company data is guaranteed to stay within the company. If an encrypted disk is brought to another outside computer the result (without the decryption key) is complete garbage. Since the encryption is automatic and established by the network administrator, the encryption keys are never seen or touched by the user. It is completely transparent to the legitimate user - only when someone tries to get outside the system will the presence of this security measure be felt.

Criteria 7: Availability

Accuracy and reliability of service are other terms which apply to availability, and may be considered fine points of distinction. This criterion's aim is to oversee the functioning of the computer in question to make sure that the system is operating as it should. Is RAM being used up too fast? Is the CPU being slowed down? Is the hard disk being filled up faster than it should be for the current application?

The availability criterion assists in identifying worm-like behaviour, Trojan horses and other items of malicious code.

We will not dwell on this criterion, but for the serious security practitioner, it provides additional protection.

The Big Picture

The reader will by now have got a feel for what security can do not only for the sanctity of the organisation's information, but also as an active deterrent against viruses and the damage they can cause.

With these mechanisms in place, the destruction wrought by a virus that successfully bypasses these barriers will be minimal to say the least, highly contained, and there will be a record of exactly what happened.

One of the most important questions is that of speed. How resource-intensive are security mechanisms that provide such dual-purpose protection?

The overheads are minimal. In 386/486 machines, the Reference Monitor occupies between 6K and 30K of (preferably) upper memory, and its effect on performance will be barely measurable - perhaps a couple of ticks on your performance test of choice (that is, not noticeable in 'real world' applications.). The encryption, depending on how well it is done, should cause less than a 5% performance reduction, and on better systems less than 3%. Again, it is very hard to notice this speed drop in the real world.

Lastly, there is the question of transparency to the user. This is a must, and I hope that it is clear that this goal has been achieved. The only intervention on the part of the user is:

- When he logs on to identify himself to the system
- When and if something goes wrong.

Other than that, if the user is using the system within the prescribed guidelines, he will not even be aware that there is any security watching his every keystroke.

Given the choice, most users prefer transparent security: as IT Managers will well know, users grow weary with the daily regimen of typical 'virus busting'. In large companies with thousands of computers, virus protection requires a level of corporate cooperation which is almost impossible to achieve, and is terribly ineffective.

The biggest advantage with security when applied to virus detection is that no periodic system-wide updates are required. Only the network administrator needs to keep one copy of his favourite anti-virus software product to check any disks which enter the system.

This would be a disaster for anti-virus vendors! Instead of tens of thousands of their ill-conceived efforts, they would need only to sell a couple to each organisation. No wonder that the 'virus busters' are not in favour of the more comprehensive and effective methods employed using security modelling!

The concept of active deterrence and containment is the key to anti-viral protection in a secure environment. Active deterrence is a three-pronged defence which directly tackles the offending viruses and their intended efforts. The three goals of active deterrence are:

1. Eliminate, at least to a high degree of probability, the chances of a virus successfully infecting a computer system. As we have shown, there are plenty of mechanisms available to keep the virus off the system. Used properly, they greatly increase network security.
2. If a virus gets through, isolation and containment are necessary to eliminate additional propagation and epidemic infection. Shrink-wrapped viruses are always a possibility. As we have shown, active containment will greatly reduce the damage caused by a virus. If a virus *does* get through, the systems administrator can use his single copy of anti-virus software to identify the virus and remove it.
3. If viruses *do* infect the system, prevent them from causing damage, identify the infected files and the users who brought the offending files into the system. In short, provide the mechanisms to track back the source and offer evidence if prosecution is in order.

Last but not least, how much will all this cost?

Site licences are available for software security products, just like anti-virus software, and many companies offer volume discount, configuration services and customisation. However, do not expect to find good security products available as shareware on bulletin boards. The good security companies just do not work that way.

Since you will not be needing to buy new licences yearly, or update pattern databases monthly, security software is a little more expensive - but as it does more, it should be. Prices range from a licence for about $10,000 to over $250,000 depending upon what you need and how many people on your network. Single copies cost from as little as $8-$10 to as much as $250.

Conclusion

Get the most 'bang for your buck'! We have little idea of where the next virus is coming from or what it will do. Passive techniques have proven to be ineffective, antiquated, cumbersome and a network administrator's updating nightmare. However, they do serve a purpose: the built-in obsolescence of the anti-virus software keeps many a vendor in business - nice work if you can get it!

VIRUS ANALYSIS 1

Evgeny Kaspersky and Vadim Bogdanov

Strange - A New Way to Hide

Stealth viruses have been around for a very long time, and are one of the principal reasons why manufacturers insist that users execute a clean boot before using anti-virus software. Many software vendors attempt to circumvent this problem by gaining 'clean' access to both INT 13h and INT 21h, the idea being that if clean disk access can be achieved, the effects of any stealth virus will be negated.

The Strange virus calls into question the logic of such techniques, as it illustrates a new way for a virus to evade detection by a scanner. By moving to increasingly low-level interception of hard disk read requests, the virus authors appear to be attempting to ensure that users who forego elementary safety precautions pay the price.

Simple Installation?

The Strange virus is a master boot sector virus three sectors long. However, here its similarity with other boot sector viruses ends. When an infected machine is booted, the virus loads itself into memory and becomes resident. The virus decreases the word at address 0040:0013, which specifies the amount of available conventional memory and then hooks INT 08h (the Timer Interrupt) rather than the 'standard' boot sector virus Interrupt, INT 13h.

The virus uses INT 08h to monitor the bootstrap procedure of the PC. When the Interrupt vector table is set up (this happens when DOS is loaded) it restores the original INT 08h handler and hooks INT 21h. The INT 21h handler simply intercepts the DOS Load and Execute function.

The rather torturous route above enables the Strange virus to intercept the loading of the command interpreter. This is done immediately *after* the device drivers are loaded. At this point the virus installs itself as a device driver and restores the original INT 21h handler. INT 13h is finally hooked, as is INT 09h (the Keyboard interrupt). If the virus is unable to install itself as a device driver, it displays the message:

```
Hmm... Strange drivers you have, very
strange... ;-)
```

At first glance this highly complex loading procedure seems completely unnecessary - after all, the virus could have picked up INT 13h as soon as the system was booted. However, there is a subtle difference between intercepting the vector now rather than at boot time. By the time the command interpreter is loaded (usually COMMAND.COM) all the relevant device drivers have been installed. Therefore any driver software required to access the DOS partition of the disk will also be installed and *already* hooked to INT 13h. This means that the virus can access the disk at a sector-by-sector level safely and reliably even in the presence of disk compression software etc.

Restricted Access

The virus carefully checks whether another program is attempting to tunnel the true INT 13h address. It does this by comparing the contents of the stack before and after a PUSH and POP instruction. While the contents of the *stack* are not altered by tracing, the contents of the memory just above the top of the stack will be, when the return address is PUSHed. If this test shows that tracing of the executable path is occurring, the virus issues an IRET with the registers containing the error code for a 'disk write-protect' error.

"even if an anti-virus program has clean INT 13h access it is still entirely capable of being 'stealthed' "

Hardware Stealth

Apart from its unusual installation process, the virus uses a previously unseen method of avoiding detection - it makes use of hardware interrupts in an attempt to hide its presence.

Whenever data is read from the disk drive, a hardware interrupt occurs which indicates that a read is ready to take place. These interrupt requests are handled differently on the XT and the AT, and therefore the first thing the virus needs to do is ascertain the processor type.

There is no built-in method of determining the processor type; *Intel* did not include any simple processor ID instruction in the i8086, and therefore no such function was built into newer processors.

The virus determines the type of processor by using five assembler instructions:

```
MOV       AX,2
MOV       CL,41h
SHR       AX,CL      ; shift right
TEST      AX,1       ; is AX bit 0 equal to 1?
JZ        xt_class_computer
```

The above example works because of a difference between the i8086 and more modern*Intel* chips. The *Intel* 80386 Programmers Reference Manual states that 'To reduce the maximum execution time, the 80386 does not allow shift counts of greater that 31. If a shift count greater than 31 is used, only the bottom five bits of the shift count are used. (The 8086 uses all eight bits of the shift count.)'

The above routine will therefore have different results when executed on an XT rather than on an AT.

The XT Stealth Routine

On an XT, the virus hooks INT 0Dh - this corresponds to the hardware Interrupt IRQ5 (the Hard Disk Controller Interrupt). Whenever a disk read is requested, the virus checks the contents of the disk buffer for its own code. If it is found, it substitutes the contents of the buffer with the contents of the original Master Boot Sector.

The AT Stealth Routine

The INT 76h handler routine is somewhat more complicated. When a disk access is about to take place the disk controller issues a hardware interrupt. This causes the virus code to be executed. On the AT, the virus checks the contents of ports 1F3h to 1F6h. These ports contain the data which the hard disk controller will use for the forthcoming disk access.

If these numbers correspond to a read of the Master Boot Sector of the hard drive, the Strange virus alters the contents of these ports so that the original Master Boot Sector is read instead.

This means that even if an anti-virus program has clean INT 13h access, it is still entirely capable of being 'stealthed'. This serves as yet another illustration of the danger of not clean-booting the machine.

Trigger

The virus contains a number of different trigger routines. Firstly, if the virus encounters an error during installation it displays a silly text message (see above).

In addition, the virus uses INT 09h to add occasional mistyped keystrokes. By far the strangest trigger however is the fact that the virus intercepts disk writes which start with the letters 'MZ', which are used to indicate that a file has an EXE format.

When the virus encounters such a sector, the disk write is allowed to pass unmolested except for the first two letters, which are swapped about. This is a bizarre action to take, as EXE files edited in this way should still function correctly, since 'ZM' is also a valid EXE file qualifier.

Conclusion

The virus is not particularly difficult to disinfect: the original Master Boot Sector is stored in sector 11 of the hard disk and can easily be copied back to its original position.

However, the way this virus uses stealth is particularly interesting, as the manipulation the virus employs in order to avoid detection is at a lower level than usual. The author of the virus appears to have an in-depth knowledge of the IBM PC and it is lamentable that a reasonably competent programmer would wish to waste his time on such a pointless (and malicious) project as this virus.

The new method of stealth does have some repercussions for those who insist that a clean boot is an unecessary luxury. Anyone advocating such a technique had better be sure that they have considered all the ways to subvert their product - or else risk users' ire when they find themselves the victim of the next crop of stealth viruses.

STRANGE

Aliases:	Hmm
Type:	Memory-resident Master Boot Sector.
Infection:	Master Boot Sector of Hard drive and Boot Sector of Floppy Drives.
Self-Recognition:	
Disks	Checks for value 047Ch at location 0124h in MBS.
Memory	Checks for value 047Ch at offset 0124h from the top of memory.
Hex Pattern:	Positioned at offset 1Ah of the MBS
	`33c0 8ed0 bc00 7c8e d8a1 1304` `50b1 06d3 e08e c026 813e 2401`
Intercepts:	INT 08h, 09h, 0Dh, 13h, 21h, 76h for installation, infection, stealth and damage
Trigger:	Displays the message 'Hmm... Strange drivers you have, very strange... ;-)', inserts random key presses, exchanges the word 'MZ' to 'ZM' if found at the start of a sector.
Removal:	Specific and generic removal is possible. Under clean system conditions replace original contents of Master Boot Sector from sector 11.

VIRUS ANALYSIS 2

Jim Bates

Swiss Army - Invading Europe's Disks

The Swiss Army virus had a somewhat confusing introduction to the virus world, as it was originally identified by *McAfee's* SCAN product as EXEBUG2. Once the initial confusion had died down, it became apparent that the Swiss Army virus was in the wild in Europe. The incident serves to highlight the problems which are caused by the chaotic naming systems in the anti-virus industry. However, it is also of some relief, as the Swiss Army virus is far less voracious than EXEBUG2.

Within the shadowy world of the virus writers, there are perhaps fewer than one in ten who actually have any recognisable programming skills. The technically capable few pose a far more serious threat than the majority of virus authors, and are obviously motivated by hatred and malice beyond our conception.

The Swiss Army virus is undoubtedly the *magnum opus* of one of the lowest echelons of the virus writers. Within this virus is a message suggesting the abolition of the Swiss Army and we might conclude that the writer has spent some time in that venerable institution. If this was so and he was as capable a soldier as he is a writer of viruses, I would strongly advise the immediate evacuation of Switzerland!

General Description

Apart from the messy and incompetent coding, this is a fairly unremarkable Boot Sector virus which infects the DOS Boot Sector of both fixed and floppy disks. The total length of the virus code (including data areas) is 1522 bytes which occupies three 512 byte sectors.

The virus is stored on the disk in such a way that the first sector will be in the normal position of the DBS. The second and third sectors of the virus code are inserted into the last two sectors on the disk (regardless of whether they were already occupied). The original DBS is written to the sector immediately before these last two. This will result in data corruption on machines where these sectors are in use.

Having shown such cavalier disregard for the fixed disk, the floppy disk infection routine contains a surprising amount of code dedicated to locating unoccupied clusters where the relevant sectors can be stored. This code is so badly written that it is difficult in places to determine exactly what the writer thought he was doing.

Installation

Analysis of this tangle is probably best begun at the installation routine. Therefore let us assume that the machine has just been booted from an infected floppy disk.

The virus code begins by accessing the base memory pointer of the machine and subtracting 3 Kbytes from it (thus making room for the virus code at the top of memory). The existing virus code (just the first sector at this point) is then moved up into high memory in a manner which highlights the inexperience of the writer.

Processing is then transferred to this high code and proceeds to hook the Disk Service interrupt (INT 13h) by direct memory access. Next, the two remaining sectors of virus code are read into memory. After this, the original Boot Sector is read into its normal location in memory.

Finally, after a check on the boot drive to see whether it was a fixed disk (this check obviously fails in this analysis), processing is passed to a fixed disk infection routine.

This routine first loads the Master Boot Sector of the first fixed disk and accesses the partition table to determine the address of the active partition. The active DOS Boot Sector is then loaded into memory and examined to see whether it is already infected. If it is, processing jumps to the original floppy boot record in memory. Otherwise, the virus code and the original DBS are written to the disk as described above. The addresses of the last three sectors are calculated from maximum values obtained from a function 8h request (get parameters) to the disk interrupt service.

Once the fixed disk has been infected, processing returns to the original floppy boot record stored in memory.

Resident Operation

As noted above, this virus connects its code into the system services by hooking the INT 13h disk services interrupt. The interception routine is one of the simplest (and yet the clumsiest) that I have seen, affecting only read requests for track zero of either head of floppy disks.

Interception begins by completing the requested function under virus control and then saving the returned register values. After checking for a 'Floppy disk removed' error, a complicated tangle of instructions is then executed to ensure that the correct flag values are eventually returned to the caller. Immediately after this, the system date is checked to see if the date is set to the 7th of February (any year). The significance of this date escapes me but if it is found, processing jumps to the trigger routine. If the trigger conditions are not fulfilled, the target floppy disk is infected and the request is returned to the calling routine.

The floppy infection routine attempts to access the File Allocation Table structure on the floppy disk and determine the location of unused sectors where the virus code can be stored. No effective check is made of the disk structure, so in some cases data on the floppy will be corrupted.

The author's intention is to mark the sectors used by the virus code as bad so that DOS will not allocate them for future use. There are usually two FATs on a floppy disk and both of them will be modified by this virus. The bugs in this section of code could well result in corruption of the FATs in a way that would make the data inaccessible.

A PC normally only accesses the DOS Boot Sector during the boot process. Thus with DBS infectors there is no technical need to redirect system requests to the original sector (although such redirection might be included in attempt to evade detection by anti-virus software). There is no redirection capability within the Swiss Army virus and therefore no special precautions are needed to locate and identify it.

The Trigger

The trigger routine is invoked if an attempt is made to read from track zero of a floppy disk when the system date is set to 7th February.

Processing here begins by decrypting a message contained within the virus and then goes on to collect various details about the system disk drives. This information is used within a comprehensive destruction routine which attempts to write garbage to every sector of every head of every track on every fixed disk!

It is interesting that a large percentage of the trigger routines that I have seen simply do not work. Perhaps the virus authors are unable (or unwilling) to test them properly and rely upon their own estimate of their astounding programming capabilities. Whatever the reason, the trigger routine in this virus certainly does not function as intended.

The distinction in this case is academic since destruction of random sectors of data will definitely take place. This corruption will occur on most partitions of most fixed disk drives on the system.

The decrypted message will be displayed after each pass of the destruction routine (counted on a drive by drive basis) and appears as:

```
Schafft die Schweizer Armee ab !
```

This, I am reliably informed, translates roughly as 'abolish the Swiss Army'. After dissecting and analysing this code I can certainly think of a better candidate for abolition!

Conclusions

This virus constitutes just another inept piece of code written by just another feeble-minded malcontent. It is quite possible that this virus originated in Switzerland which is rumoured to be the birthplace of the Form virus and is certainly that of the Tequila virus.

The authorities in Switzerland have interviewed known virus writers and as well as taking no further action have steadfastly refused to share information with law-enforcement agencies in other countries. If the Swiss are attempting to emulate Bulgaria as protectors of virus writers, it is reasonable to suggest caution to all computer users in dealing with or through computer technology in Switzerland.

Countries across the world are gradually waking up to the damage and loss that computer viruses can cause and are enacting legislation designed to bring justice to the perpetrators. The Swiss may wish to maintain their reputation for insularity and if so, that is their affair - as is any consequential loss of confidence. However, the advent of freely available Internet access has made the world a very small place - how long until the Swiss are forced to cooperate?

Swiss Army

Aliases: None known.

Type: DOS Boot Sector infector

Self-Recognition:

Disks Word at offset 93h in the DBS has the value 368Dh.

System None

Hex Pattern:

```
AC01 0074 03EB 9103 E9B9 005E
5859 C38D 365A 019C 2EFF 1CC3
```

Intercepts: INT 13h - on READ of any floppy, track zero. Checks for trigger date, triggers or infects floppy.

Trigger: Date is 7th February any year. Overwrites random sections of all local fixed disks on system.

Removal: Replace DOS Boot Sector with original stored on last physical sector but two of the first fixed disk drive

VIRUS ANALYSIS 3

Necropolis Returns from the Dead

The Necropolis virus has been known about for over a year, but until now has not been observed in the wild. The code is highly complex and well written - from its writing style one would guess that this virus is the work of the so-called 'Dark Avenger'. Once again, however long this 'brilliant programmer' took to develop Necropolis, it was disassembled (by a far from brilliant analyst!) in only a fraction of that time.

This virus uses highly complex stealth techniques in its attempts to subvert certain types of virus detection software. It is a simple prepending parasitic virus which infects both COM and EXE files. The virus operates by copying 1971 bytes of program code from the beginning to the end of a target file and then prepending the virus code. The code itself is 1963 bytes long excluding a static data area, and contains no trigger or payload routines.

Installation

Necropolis is a resident virus and the first time it is invoked the following sequence of events takes place:

The code first checks the version of DOS in use and if it is earlier than 3.0 immediately exits. A check is then made for the existence of the multiplex function (INT 2Fh) and if this is not present processing also returns to DOS.

The virus continues by shrinking the allocated memory to 5120 bytes before resetting the stack position and executing a routine designed to trace the DOS origins of the Disk and System services (INT 13h and INT 21h). This process has become known as 'Interrupt Stripping' or 'Tunnelling' and here it is taken to logical extremes.

Internal services are used to locate the segment that contains the DOS code and this is used in conjunction with the single step interrupt to locate the DOS and BIOS entry points.

The virus does not directly use INT 13h. However, this interrupt is located simply so that all disk acesses made can be redirected around any monitoring software that may be resident in memory. The DOS routine is examined carefully to determine whether any additional service has been hooked into it and if so, the virus intercept routine is connected 'underneath' it so that DOS access can be achieved without alerting any monitor. The connection process automatically excludes the possibility of the virus code being installed twice, since the code examination during subsequent attempts at installation will fail to find the requisite address. This rerouting will undoubtedly cause tremendous problems on some types of access control system, since the reference monitor has been effectively unhooked by the virus code, leading to unpredictable results.

After installation, the virus collects the name of the host program from the environment and proceeds to load and executes it in newly allocated memory under virus control. If the environment location is hidden or subverted (a process adopted by some resident software), the virus switches to a different routine which loads and executes the host file by reference to the internal DOS file tables. After execution, control returns to the virus code which completes some small housekeeping tasks before returning to DOS.

Operation

Once resident, this virus maintains extensive connections to the system services available through DOS. All these services are associated with INT 21h and they are handled in a number of different ways to ensure both stealth and virus replication. In order of their occurrence within the virus code, the intercepted functions and the associated action on the part of the virus are as follows:

Function 48h - Allocate memory
Function 4Ah - Re-allocate memory
Function 4B03h - Load Overlay

The virus takes control of the current program segment and completes the requested function under virus control. The program segment is then freed and processing returns to the calling program. This prevents the virus code from being overwritten in memory since it normally occupies a section of free memory.

Function 31h - TSR request
Function 4Ch - EXIT request

Here, the virus sets up a loop of instructions which attempts to infect any open files that have an extension of COM or EXE. After this loop, processing reverts to DOS with the function request.

Function 0Fh - FCB Open
Function 10h - FCB Close
Function 17h - FCB Rename
Function 23h - Get File Size

The File Control Block functions are a throwback to earlier versions of DOS but are nevertheless used often by a lot of software. The virus collects the target filename from within the function request parameters and proceeds to infect it if it is COM or EXE. Processing then jumps to the requested DOS service routine.

Function 3Fh - Read File

If the file has a COM or EXE extension, it is infected and the read request is completed under virus control. If the newly read data in memory contains virus code, this is overwritten by the original code The location in the file where the read was made is then checked to see if it contains any virus code and if so, this is replaced by the original code that occupied that position before infection. Thus to inspection by this function, an infected file appears clean.

Function 3Dh - Open File
Function 43h - Change file attributes
Function 56h - Rename a file

If the target file has a COM or EXE extension it is opened, infected and then closed. Processing then jumps to DOS for completion of the original function request.

Function 3Eh - Close File

The virus infects the target file if it has a COM or EXE extension before continuing to DOS with the original function request.

Function 14h - FCB Sequential Read
Function 21h - FCB Random Read
Function 27h - FCB Random Block Read

In this case, the treatment is similar to the other FCB requests noted above, but if the infection was successful, the stealth routines are invoked to conceal the infection after the Read request is completed under virus control.

Function 4B00h - Load and Execute
Function 4B01h - Load but don't Execute

The last two subverted functions read the whole file and infect it regardless of its extension. The infected file is then repaired before either returning to the calling program (Function 4B01h) or being executed (Function 4B00h).

File infection only takes place if the file is not already infected, if it is not a system file (determined by the attributes) and if it is greater than 31 bytes in length. Multiple infection is prevented by comparing a substantial part of the file code (195 bytes at offset 235) with the similar section of virus code. An additional check generates a checksum of the whole virus and compares it to a similar checksum written to the data area just beyond the end of the virus code when the file was infected.

Throughout all this interference with system services, the Necropolis virus uses many undocumented functions to gain unhindered access to the files on disk. System File Tables are subverted to allow write access during infection without alerting any potential monitoring software. The low level BIOS services are rerouted directly to DOS (INT 13h) and ROM (INT 40h) to prevent possible monitoring during low level writes and then repaired immediately after use. Even the termination routine which DOS uses to regain control after program execution, is subverted by the virus.

Although this is one of the most comprehensive stealth viruses currently at large, there are some obvious holes in its security which most reasonable capable anti-virus software should be able to use with ease. The most obvious of these, as always, is that no stealth capability can hide virus code on a machine that has been rebooted 'clean'.

Conclusions

All virus code that crosses one's desk is saddening and frustrating, but when it is obviously written by someone who displays some skill and experience as a programmer it is also difficult not to feel extremely angry!

Continuing rumours suggest that there are people who know who the 'Dark Avenger' is. If so, they should publish his name (and the proof of his identity) so that he can no longer hide behind his precious sobriquet and we can all treat him with the contempt he so richly deserves. Far from being 'brilliant' this person is perverted, he continues to blight the future of the computer industry.

NECROPOLIS

Aliases:	1963
Type:	Prepending Parasitic file infector
Self-Recognition:	
Files	Checksum not as expected.
Memory	Virus fails its own installation routine.
Hex Pattern:	
	`2EC7 0606 09AF 08B4 01FF 1E4C` `009D 2EC7 0606 09AB 08B4 0BFF`
Intercepts:	INT 21h many functions - for infection and stealth
	INT 13h, INT 23h, INT 24h, INT40h temporary for stealth
Removal:	Disinfection is possible but best to delete and replace infected files under clean system conditions.

PRODUCT REVIEW 1

Mark Hamilton

PCVP - A Panacea for all Ills?

Computer Security Engineer's PC Vaccine Professional (*PCVP*) is the latest anti-virus package to be reviewed by*VB* and is the third product that I know of to include the word 'Vaccine' in its name. If this trend continues, it will soon be impossible to distinguish each product from its neighbour!

Sapristi! On Nous A Oubliés!

PCVP is available on either 3.5 or 5.25-inch permanently write-protected media and comes with a slim, 67-page A5 sized manual. *CSE's* manual starts with an introductory remark: 'Because of the unlimited number of ways to write a virus, *PC Vaccine Professional* is probably not able to detect and protect against all types.' How refreshingly honest; I wish more companies were as open and straightforward with their users rather than bombard them with meaningless hype.

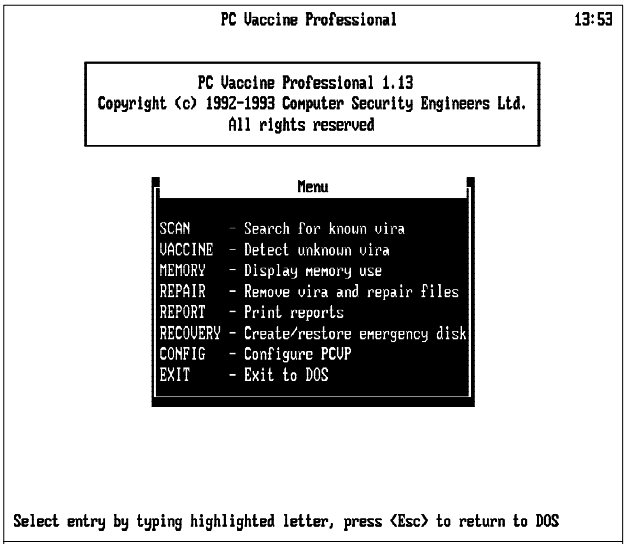
PCVP's diskette contains about 387 Kbytes and includes an installation program. When this program is run for the first time, the user is given a choice of language in which he would like to operate the product. The version reviewed had options for Danish, Dutch, English or German. Being somewhat unadventurous by nature, I opted to install the English version of the product.

Unusually for an anti-virus package, there is no attempt to check memory or pre-scan the destination drive for viruses as part of the installation process. This could be viewed as a serious oversight on the developer's part, though in fairness, the manual does suggest rebooting the PC with 'an original write-protected DOS diskette'.

The installation routine copies across approximately twenty files from the master disk. This comprises the selected language-specific version. However it ignores some 14 files from the KILL sub-directory on the master disk. This directory contains various virus-specific disinfectors for common viruses such as Michelangelo, V-Sign, and Form - these need to be run from a write-protected disk.

Software Which Can Learn

PCVP includes a device driver which occupies some 6 Kbytes of memory. It does not look for specific viruses but instead warns the user if a program starts to exhibit virus-like behaviour such as attempting to write directly to the disk by bypassing DOS.



PCVP has all the usual features of an anti-virus product. However, the implementation detracts from an otherwise good scanner.

The problem with this type of approach is that there are so many programs whose perfectly legal activities would be trapped by this method, making it prone to false positives.

How does *CSE* get around this? Easy - it has included an option to 'teach' the product that a particular operation is legal. This is done by pressing Control-L whenever its intercept banner appears. However, I strongly question the logic behind this approach.

In order to use this mode correctly, you need to be *absolutely* sure that the warning is a false positive. Most users simply do not have the technical expertise to make this kind of decision - it is likely that after a couple of false alarms they would automatically respond with the 'okay' keypress. This seems like an administrator's nightmare - if this option should exist at all, it certainly ought to be possible to disable it.

The program interfered with*Windows* on one test machine, causing it to crash rather unceremoniously. I was not able to pin-point the cause, but the problem was reproducible.

Generic Checks

PCVP.EXE is the main part of the package, and allows the user to scan drives, check files for changes, display memory and so on. One immediate problem I did encounter was that *PCVP* assumes that all the drives it detects are available and if they are not, causes a DOS error message to appear. The test machine happened to have a 1 Gbyte optical drive attached, and the program would not proceed until the optical drive had been mounted.

The generic checker allows the user to ensure that no changes have been made to files. The integrity database which *PCVP* creates is encrypted, although there is no mention of what algorithm is used. If the checksum database is deleted, the user is alerted and asked whether he would like to take a fingerprint of the disk.

When *PCVP* discovers an altered file, it uses heuristic analysis to determine whether the changes are due to the file being updated or whether the changes have been caused by a virus infection. This controls the number of false alarms produced by integrity checking packages.

All the results from the integrity checker (and all other parts of the package) are sent to a log file. There is no built-in mechanism for viewing this file - it has to be printed. As above, *PCVP* did not handle any errors encountered (such as the printer being off-line) gracefully - the user is immediately confronted with the familiar DOS 'Abort Retry Fail' prompt.

Detection Results

The scanning engine is the strong point in *CSE's* product, being both accurate and quick. *PCVP* detected all samples in both the Mutation Engine test-set and the 'In The Wild' test-set. Its results in the *VB* 'Standard' test-set were similarly impressive, as it missed only one virus. This accuracy is not at the cost of speed however: *PCVP* took just over 15 seconds to scan the test hard drive - a creditable result.

All the options in the scanner can be enabled or disabled by typing the first character of its name. Unfortunately, two options share the same letter: Memory and More. Hitting the 'M' key alternatively enables and disables More - which pauses the screen display when full. There is no way, without using a mouse, to disable memory scans.

One annoying fact I discovered is that while you can selectively enable or disable the scanning of certain files, there is no mechanism for scanning all files or adding to what the developers consider to be executable code.

Conclusion

I am left with the feeling that this software was rather rushed into production. What is there works and it does have impressive virus detection capabilities. However there has been a distinct lack of attention to detail, for example the inability to disable memory scans from the keyboard, the fact that the DOS error trap is used, the problems encountered with the optical drive and so on.

Provided the company can sort out these glitches and keep to its promise of providing a monthly update service, then *PCVP* may well stand some chance of success in this already over-subscribed market.

PCVP

Scanning Speed

Hard Disk:	
Turbo Mode (1048.9 Kbytes/sec)	15.4secs
Secure Mode	N/A*
Floppy Disk:	
Turbo Mode (47.8 Kbytes/sec)	6.3 secs
Secure Mode	N/A*

Scanner Accuracy

'*VB* Standard' Test-set[1]	363/364
'InThe Wild' Test-set[2]	116/116
'MtE' Test-set[3]	1536/1536

Technical Details

Product: *PC Vaccine Professional*

Version: 1.13

Author: *Computer Security Engineers Limited*, New St James Place, St Helier, Jersey JE4 8WH. Channel Islands

Telephone: 0534 500400

Fax: 0534 500450

Price: £18 per PC (minimum charge £900)

Test Hardware: All tests were conducted on an *Apricot Qi486* running at 25Mhz and equipped with 16MB RAM and 330MB hard drive. *PC Vaccine Professional* 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 10 files of which 6 (310,401 bytes) were executable.

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

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

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

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

*Editor's Note: *PCVP* does not have two distinct 'Secure' and 'Turbo' modes. However, the mode of scanning used by *PCVP* is more akin to other products' 'Turbo' modes, and therefore its times have been shown as such.

PRODUCT REVIEW 2

Dr. Keith Jackson

Victor Charlie - Anti-virus Ninja Warrior

This month's review looks at an anti-virus product that is *very* different from the endless scanners and checksum programs that form the core of *VB's* reviews. These differences are obvious before the product has even been taken out of its packaging; the disks and manuals arrive inside a small cloth bag with *Victor Charlie* (*VC*) advertising material plastered all over it. I can only describe it as being rather like a ladies handbag (or purse if you are American!).

Victor Charlie claims to be 'the world's first generic anti-virus defence', which 'protects your files, critical system areas and data against all viruses'. It also claims to require 'no continuous updating', and to have a 'non-technical manual'. All these quotes are taken from the outside of the *Victor Charlie* 'handbag', so any user is going to see them immediately. It all sounds too good to be true. Is it?

Documentation

Victor Charlie comes with an installation guide, and a reference manual, which uses a pseudo-military style throughout, and certainly lives up to its claim of being non-technical! The anti-virus programs are referred to as soldiers going out to conquer the invading enemy (viruses). For example, the documentation claims that the software is 'waging war on viruses'. To achieve this, its major components are referred to as 'shock troops' which are 'ordered out on patrol' (executed), and then 'invite ambush by a virus'.

When these files detect their 'lurking enemy', this is said to be because 'an active virus is by its nature unable to resist the urge to try to attack' *Victor Charlie's* anti-virus programs. The anti-virus programs detect this event, extract a signature, 'kill' the virus, and then 'commit suicide', 'Just before they die', the *Victor Charlie* anti-virus programs produce the following daft message 'VC CAUGHT A VIRUS FROM YOUR MACHINE! (SUICIDING NOW)'.

I could go on *ad infinitum* providing quotes to show that this nonsense pervades the documentation, but the pieces of text within quotation marks above are reprinted verbatim, and provide a representative sample of the style used. Note that this is supposed to be a reference manual!

How anybody could think that documentation should be written in this way is beyond me. The childish style employed is completely unnecessary, and the marketing personnel who dreamed this up have done the sales of *Victor Charlie* no favours whatsoever. If it were just a marketing ploy, and real technical detail was hidden in some corner of the documentation, then I could possibly be persuaded to overlook it, but in this case it seems to be used to hide the fact that the manual is almost bereft of hard content.

Installation

Having criticised the style of the documentation, and the lack of technical content in the reference manual, the sole redeeming feature of the documentation is the short (15 pages) installation manual. It explains the difference between a quick installation (where *Victor Charlie* makes all the decisions), and a custom installation where the user must specify the sub-directory in which the *Victor Charlie* files are stored, and list the files (up to a maximum of 10) which are to be protected specially against virus attack.

At the end of installation, the software offers to calculate a 'bitcheck' list for all of the executable files on the hard disk, which it uses to detect any changes made to these files.

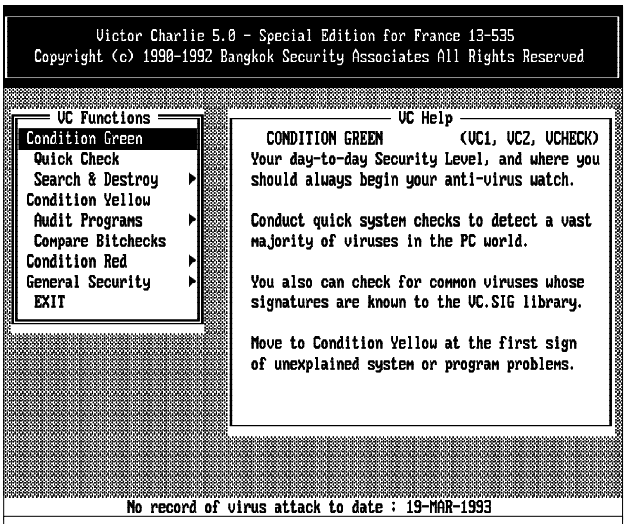
Note the new piece of jargon introduced here - a 'bitcheck', which is claimed by the developers of *Victor Charlie* as a trademark. The developers claim that 'bitchecking' employs 'proprietary algorithms to create a special, double-encrypted number for each file'. From the information provided in the reference manual, users cannot judge whether this 'bitchecking' process is any good. What algorithm is used to calculate the checksums? What algorithm is used for encryption? Why *double*-encrypt - is a single encryption insecure?

Although the documentation does not mention installation to a floppy drive, I tested this out. The individual components worked correctly when installed on a floppy disk, but the shell program just hung, and presented the error message 'Divide Overflow' when used in this manner.

Condition Green

When installed, *Victor Charlie* consists of several components which can either be executed individually, or can be executed by making a selection using a shell program. This shell program provides protection at three distinct levels.

'Condition Green' (the lowest level) has two selection options, one of which performs a 'Quick Check' of the computer, and the other (entitled 'Search and Destroy') looks for viruses on all selected parts of a disk. If anything suspicious is detected, the user is told to go to 'Condition Yellow', which adds a program to perform an 'audit' (calculate a 'bitcheck' for each file being protected), and a 'bitcheck' comparison program.



The entire product is filled with pseudo-military jargon - an approach which seems better suited for a computer game than for a serious computer security product.

If virus activity is detected, the user is advised to move to the 'Condition Red' options where utilities are provided to repair the boot sector and the partition sector, to remove viruses from floppy disks, to make a Rescue Disk, to view the log files produced by *Victor Charlie*, to turn off false alarms, and to see various demonstrations of virus-like activity.

I'm not exactly sure in technical terms what the two 'Quick Check' programs do (two separate programs are invoked, twice each), as the documentation is quite vague on this point. Claims are made that these programs check the 'system' (whatever that means), computer memory, and 'other vital elements'. This is expanded later in the documentation to include checks on the Boot Sector, the Partition Sector and the System Files, but nowhere does the documentation explain precisely what these two programs are doing.

On-the-fly Detection

I tested out detection of virus activity by carrying out a 'Quick Check', infecting my test computer, then performing another 'Quick Check' to see if the virus infection was detected. Each test took about 15 minutes per virus tested, so I only tested a random sample of 10 viruses chosen from the 186 in my test-set. Four of these viruses were detected by *Victor Charlie* as being resident on disk when a scan for the signatures of 'common' viruses was invoked (Dark Avenger, Alabama, Cascade and Fish6). The other 6 virus samples were undetected by *Victor Charlie* during a scan - Amoeba, Burger, DataCrime2, LeHigh, Icelandic1 and Kamikaze.

Using the above described test, *Victor Charlie* failed to detect active execution of 4 of the 10 samples; these were Alabama, DataCrime2, Icelandic1 and Kamikaze. The results of these tests are hard to interpret with any certainty, and would probably become no clearer if all 186 samples were tested as there are so many variables involved. However they do show clearly that some viruses can access the boot disk without being detected by the 'Quick Check' process. This is unsurprising, but it is not what the documentation implies. However, without hard technical details of the product it was difficult to test it further.

The other *Victor Charlie* component which is available at the so-called 'Condition Green' level is called 'Search and Destroy'. It conducts an anti-viral scan of an entire disk, selected subdirectories or even selected programs. This program checks that the size and the 'bitcheck' of each file is still correct, and needs some minutes to work its way through an entire hard disk. It can carry out a conventional scan of a disk, but this option is not activated by default.

To give some idea of the relative execution times, the 'Quick Check' utility took just 7 seconds to test out the hard disk of the *Toshiba* 3100SX described in the *Technical Details* section, whilst the 'Search and Destroy' utility took 3 minutes 39 seconds to test out the same hardware.

I tried to fool the 'bitcheck' testing process by making single bit alterations to a series of test files, but *Victor Charlie* successfully spotted every alteration. It also spotted all files that had disappeared, and all new files. I was quite impressed by these 'bitcheck' test results, but to be convinced of their veracity, some technical details of the 'bitcheck' process need to be contained in the documentation. It is not good enough to hide behind a trademark.

Scanner Accuracy

When a scan is invoked which actually executes a virus-specific search, *Victor Charlie* claims to look for the signature of common viruses, and for signatures added to this list by the 'Quick Check' routines (see below). This sounds good, but when tested against the 186 virus samples listed in the *Technical Details* part of this review, only 47 test samples were detected - a poor result. No Mutation Engine samples were detected - a result which I believe would not be improved by any amount of automatic signature extraction!

No doubt the developers of *Victor Charlie* would defend these results on the grounds that many of the samples are not 'common' viruses as defined by Patricia Hoffman. I disagree strongly with such an approach, and all the poor souls who have been infected with a virus that is not classified as being 'common' will no doubt agree with me. In reality, scanning merely for common viruses seems a neat way of avoiding the increasing amount of work that is necessary to keep up with the ever-expanding total number of known viruses.

Victor Charlie took 3 minutes 32 seconds to scan a hard disk containing 270 executable files spread over 10.2 Mbytes. For comparison purposes,*Dr Solomon's Anti-Virus Utilities* performed the same scan test in 40 seconds, and *Sweep* from *Sophos* took 4 minutes 49 seconds for a complete scan, and 59 seconds in quick scan mode.

Condition Yellow

At 'Condition Yellow', an option is available which unlike 'Search and Destroy' just performs a 'bitcheck' audit (nothing else). This took 3 minutes 30 seconds to test out the same hard disk.

I am at a loss to explain why a utility which performs fewer tests than 'Search and Destroy', but utilises the same 'bitcheck' test as the core of its testing, is available at a higher level. I must be missing something, but I've been through this about three times and I still don't understand it.

The utility programs provided for use at 'Condition Red' seem to operate as claimed, and all except the utilities used to remove viruses are disabled when a virus is detected by *Victor Charlie* - a nice touch.

Painless Extraction?

I've been quite harsh on this product up until now, but careful readers will note that most of my objections have been against the lurid phraseology employed, and the lack of technical detail in the reference manual. However when it comes to automatic detection of virus signatures, and false alarms in general, I have a real bone of contention.

Whenever Victor Charlie finds a virus which is not detected by its scanner it attempts to extract a scan string for the virus. It is obvious that the developers of*Victor Charlie* are quite proud of their automatic extraction of signatures. They have clearly studied the problem of the many possible false alarms that can result from using this process, and have still decided to use it. Indeed, an option is provided to disable false alarms, and the reference manual states quite clearly that, 'In the event that VCHECK causes a false alarm on one, two or three programs, you should consider living with these'.

False positives are probably the biggest single problem facing anti-virus developers today, and to consider that signature extraction can be performed automatically when researchers are spending many man hours studying each virus for a reliable signature, is almost certainly ill-advised.

Put another way, if automatic extraction of signatures actually does work, why don't all the anti-virus software developers just run their virus libraries through the automatic detection process to detect a reliable signature? How can an automatic signature extraction process deal with self-encrypting viruses, or with polymorphic viruses?

Conclusion

I cannot end this review without one more reference to the militaristic style employed by*Victor Charlie*. The initials *VC* were used to refer to the Viet-Cong during the Vietnam war, and when the same initials are used throughout a product intended to fight against computer viruses, in a pejorative manner, I find such connotations distasteful in the extreme. Measured against an imaginary scale of importance, computer viruses rank nowhere against the unnecessary mayhem perpetrated during the Vietnam war.

It is the absolute certainty which*Victor Charlie* applies to the detection of viruses that I rail against most. The documentation even states quite boldly that 'No virus ever can be written for the DOS operating system which can evade detection by bitchecking'. It is a brave soul that makes such a claim, and it is a foolhardy user that believes it.

Such extravagances are a shame, as lurking within*Victor Charlie* is software that*could* be very useful indeed, but I recommend avoiding this product until the documentation permits *Victor Charlie's* strengths and weaknesses to shine through the fog induced by the marketing-speak used in the manuals. Then the user can make his own mind up about the facilities on offer.

Technical Details

Product: *Victor Charlie*

Developer: *Bangkok Security Associates*, 888/32-33 Ploenchit Road, Bangkok 10330, Thailand. No telephone number or fax number provided.

Vendor: *Vecteurs Technologies*, 2 Place de la Defense, CNIT BP 240, 92053 PARIS - LA DEFENSE, France. Again, no contact telephone number or fax number is provided.

Availability: IBM PC/XT/AT or PS/2 running MS-DOS 3.0 or above (not DR-DOS), with a minimum of 256 Kbytes of RAM for each standalone program, and a minimum of 512 Kbytes of RAM for the menu-driven integrated shell program.

Version evaluated: v5.00

Serial number: Special edition for France, 13-535

Price: 650FF

Hardware used: (a) *Toshiba* 3100SX, a 16MHz 386 laptop, with 5 Mbytes of RAM, one 3.5 inch (1.44M) floppy disk drive, and a 120 Mbyte hard disk, running under MS-DOS v5.0. (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

For details of the test-set used refer to *Virus Bulletin*, December 1992, p.22

INDUSTRY WATCH

Share and Share Alike

March has been a troubled month for anti-virus supremo John McAfee. In last month's edition of *Virus Bulletin*, it was reported that Version 100 of *McAfee's* CLEAN program had a bug in it which could cause corruption of hard drives. To make matters worse, it was the disinfection routine of the Michelangelo virus (set to trigger on March 6th) which was affected.

The bug is claimed to only occur on '1 in 500' PCs - if this is true then *VB* was extremely unlucky: two of its test machines were affected by the error. In both cases, the hard drive was unreadable, and data recovery was not trivial.

Serious bugs like this must be reported to users as soon as they are known - however, things did not happen that way in this case. *VB* knew about the bug well before the information was posted to the internet usegroup comp.virus, and it is likely that *McAfee Associates* knew of it well before *VB*. Many users of SCAN were not informed of the problem at all, and continued to use CLEAN v100 until it was updated to version 102. When bugs of this magnitude are found, users have every right to expect to be informed immediately - not passing on full details quickly must surely be considered negligent.

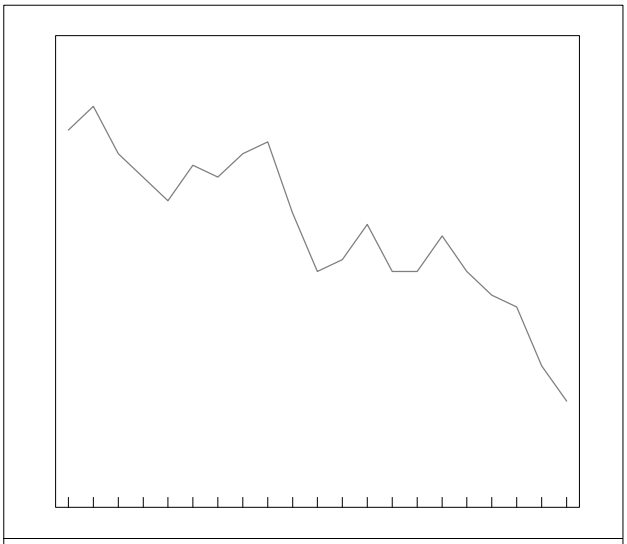
Courting Trouble

Another hurdle which the company must overcome this month is the impending suit with *Imageline* concerning a false positive found by SCAN in one of *Imageline's* products. *Imageline*, having won the 'first round' in this legal wrangle, is now looking for 'substantial' damages from *McAfee Associates*. At this time, no other details of the case are available, but it is currently being tried and full results will be given in next month's *VB*.

That Sinking Feeling...

The controversy surrounding *McAfee Associates* has hit the company hard where it hurts most - in the wallet. *The Wall Street Journal* (March 1st 1993) carried a four column article on the fortunes of the company. Stating that the 'Nasdaq-traded shares of *McAfee Associates* have been acting as sickly as an infected PC' the article went on to review the problems which *McAfee Associates* faces.

A recent trend on the stock market has been a loss of faith in technology stocks. Even before the latest problems over the bug in CLEAN, *McAfee's* stock had been taking a battering



McAfee Associates week closing share prices (in dollars) for the period October '92 to February '93

on the stock market as investors shied away from such 'risky' investment in favour of companies with a broader array of products on which to fall back.

The shares have recently tumbled to an all time low of $4 3/4 from a high of $23, as the two investment banks which took the anti-virus software vendor public last year reduced their 1993 earnings forecast of the company. *Alex Brown & Sons*, a co-underwriter of the *McAfee* IPO lowered its estimate by 5 cents a share. W. Christopher Mortenson, an analyst for the bank, also stated that he had downgraded his rating of the stock from 'buy' to 'hold'. Mortenson says that his estimate reductions reflect lower expectations for booking for new licences and licence renewals.

As You Sow

It is highly ironic that it is the Michelangelo virus which is contributing to the problems which *McAfee* is suffering. The scanning frenzy which followed last year's Michelangelo publicity certainly helped *McAfee's* sales: 'Part of this (earnings estimate cut) is just saying "hey, this market is not growing as quickly as we thought" '. Mortenson explained. He went on to add that the impact of the virus on *McAfee's* sales 'was greater than expected'.

Whether the slip in the *McAfee* stock price is simply a temporary dip waits to be seen - either way, investors in the company have seen half their money disappear. Many pundits in the anti-virus industry feel that the story is one of poetic justice: had *McAfee* not hyped the Michelangelo virus, the fact that the CLEAN routine failed would have passed virtually unnoticed.

END-NOTES AND NEWS

According to a recent Home Office report, many **Fraud Squad detectives do not regard computers as potential sources of evidence**. The report found that although 85% of UK Fraud Squad detectives encounter computers during the course of investigations, none have received adequate training in how to use them. Sergeant Micheal Guinney, a member of the Merseyside Fraud Squad said, 'Police chiefs are frightened of computers, and that means that mistakes are being made because of ignorance of IT'.

Central Point Software* has launched *PC Tools for Windows, which includes enhanced versions of *Central Point's Anti-virus for Windows* and *Central Point Backup for Windows*. The product also includes the astonishingly safe-sounding DiskFix - a module which allows users to schedule automatic repair of common hard disk problems from within *Windows*. For further information contact Diane Paternoster. Tel. 081 848 1414.

Datawatch* has slashed the price of *Virex PC from $99.95 to $49.95. Other changes include free electronic updates via *Datawatch's* in-house Bulletin Board System. The products have also been repackaged, thus completing the transition of the previously acquired *Microcom Inc.* products. Tel. +1 (919) 490 1277.

Symantec appears to be getting twitchy as the impending launch of MS-DOS 6 draws nearer. Always preoccupied with the well-being of the public's PCs, it has warned that if users want complete protection against computer viruses, they will need to look carefully at what MS-DOS 6 has to offer.

Copies of *Hoskyns' Project Manager Workbench* software were despatched on disks infected with the Form virus. *Hoskyns* has sent out anti-virus software and clean replacements to every recipient of the infected disks. While shipping the infected disks is regrettable, *Hoskyns* deserves praise for 'coming clean' and the swift action which it took. Tel. 071 734 2660.

'Computer hackers cost US businesses as much $60 million in 1991', said Thomas Eaton, manager of corporate information and network security for Dayton-based *NCR Corp*. Eaton added that *NCR* has developed a computer security system that has thwarted hackers and prevented intrusion by major viruses.

Steve Jackson has won his long-running case against the US Secret Service. The Jackson Lawsuit revolved around the seizure on March 1st, 1990, of computer equipment from *Steve Jackson Games* by the secret service. Although charges were never filed against the firm, the equipment was held for some months, which Jackson claims severely disrupted his operations. Mike Godwin, legal services counsel, issued a statement saying that, 'Judge Sparks has made it eminently clear that the secret service acted irresponsibly. This case should send a message to law-enforcement groups everywhere that they can't ignore the rights of those who communicate by computer.'

S&S International's ''leading specialist virus and security publication'' *VNI* has announced a one-day conference, to take place at the *Sheraton Skyline Hotel*, Heathrow on 23rd June 1993. Precise details of the programme have not yet been revealed, but the highlight of the event is expected to be a rare personal appearance by the reclusive Doctor Alan Solomon.

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