June 1992 ISSN 0956-9979

# VIRUS BULLETIN

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



## CONTENTS

# EDITORIAL

## Profits and Prophecies

The milk of human kindness does not flow freely in the veins of the computer virus industry. Where profit can be made it will be made. Every time there is a 'crisis' (à la Michelangelo) the industry sheds crocodile tears at the fate of the 'poor computer user' and laughs itself all the way to the bank. Let there be no doubt; the efforts of most anti-virus product developers are motivated not by love but by *money*.

Not that there is anything intrinsically wrong in the profit motive - *it gets things done*: those bordering on the clinically insane may find intellectual solace while wading through 10K of Whale excrement or unravelling mutating spaghetti, but most well-adjusted personalities will only undertake this sort of nonsense for hard cash. The few eccentrics who dedicated themselves to pure virus research have ended up, by necessity, involved in software development and its attendant commercial warfare. Virus research *per se* does not pay the rent.

Early expectations that the virus problem could be cured by the charitable and benevolent intervention of the selfless few have withered and died. UK readers may remember the trials and tribulations faced by the members of *CoTRA*, the erstwhile *Computer Threat Research Association*, as they tried to form a cohesive, non-profit-making body to combat viruses and other menaces. The organisation was disbanded within months, due largely to apathy; there was simply no commercial motivation or financial reward to be had.

Over the years, certain individuals have changed their minds regarding the charitable status of remedial software. David Stang, former research director with the profit-driven *NCSA* and now director of the equally commercial *ICSA* has in the past proclaimed that PC users don't deserve viruses and should not, therefore, have to pay for remedies - he now produces publications on the subject costing up to $395.00 while asking those who write for him to forego payment! Others have changed their minds regarding the status of such software and its value - readers with long memories may recall the declaration by one high-profile vendor that anti-virus software was 'snake-oil'. The same individual later stated that no anti-virus software package should cost more than £5.00; his software currently sells for £95.00 - this 1900% increase in Recommended Retail Price compares interestingly with the UK's annual rate of inflation.

In fact, there are very few disciples of free anti-virus software still breathing. The Macintosh community is lucky to have the services of the likes of John Norstad, Chris Johnson and Jeff Schulmann, all of whom actively support public domain anti-virus software. Conversely, there are hardly any significant public domain anti-virus packages for MS-DOS; a fact that is probably accounted for by the sheer pace of viral developments on the PC over the last few years. To keep a virus scanner alive and well requires hundreds of man-hours of work each and every month; no-one is going to undertake this work unless they get paid for it. In all cases the remuneration for this effort will exceed the sustenance level and in certain cases it will result in a maximised profit for the developer.

Maintaining virus scanning software is a relentless, ongoing struggle - the R&D equivalent of an infinite loop. It is a struggle which consumes ever more man hours as the viruses become more sophisticated and the means to search for them more involved. If we have learned nothing else in the last few years we should at least understand that this work will not happen for free; the end-user must accept the burden of payment as part and parcel of using a computer in the 1990s.

The profit motive will ensure that scanning software is supported until the point is reached at which it is no longer profitable to continue development. Apart from the limitations of disk space and run-time, a third and decisive factor will ensure that many companies discontinue virus-specific software in the foreseeable future:

THE LAW OF DIMINISHING RETURNS

The PC virus industry, which comprises approximately 100 commercial outfits worldwide, can explore a *maximum* market totalling about 100 million PCs. These figures do not imply that each manufacturer is guaranteed a universe of 1 million PCs - market positioning must be accounted for. Prominent companies will take the lion's share of the market while lower-profile companies fall by the wayside as the effort to keep pace with developments exhausts programming resources - the cost of the R&D effort to maintain their products will outstrip the revenues that these products generate.

*Technical competence alone will not provide a life-raft*; sound management and a sensible pricing structure will determine which companies sink or swim. Some companies, in their desperation to win orders, have cut software prices to the bone and thus lumbered themselves with costly update commitments not offset by incoming revenues. Expect some shocks in the future as technically accomplished companies falter due to such poor management and planning.

It looks set to be a bleak future. Three years from now, there will be far fewer virus scanners available. Some packages will be properly supported from the established industry giants, others will sell in volume, not as result of their efficacy, but due to high-profile marketing. The dogged determination to 'Carry On Scanning' among some vendors is admirable. Dr Solomon, for example, has declared his intention to 'be the last off the cliff' and quotes Churchill on the subject of the British Bulldog. On the supposition that 'Dark Avenger', in a fit of uncharacteristic chivalry, decides to discontinue development of his 'Mutation Engine', this scanner development will continue unabated. But what happens when the number of viruses reaches 10,000? 100,000? 1,000,000?

# TECHNICAL NOTES

## A Case of Mistaken Identity

One of the viruses reported this month is named Gotcha-E, as it is clearly related to the four previously known members of the Gotcha family. This virus has one highly unusual feature - it attempts to demonstrate the inaccuracy of certain virus scanners. This is done by including selected fragments from various other viruses in the virus' body. The virus seems to be targeted against the *SCAN* and *TBScan* programs. *SCAN* detects the following viruses in infected files:

Datacrime
Datacrime II-b
Syslock
Tiny 133
Enigma

*TBscan* reports a slightly different set of viruses:

Datacrime IIb
Syslock-related
Datacrime B
Old Yankee-related

Other scanners which were tested performed somewhat better - IBM's *VIRSCAN* only reports three viruses (MILOUS, SYSLOCK and DATACRIME-1280), but if detection of mutants is enabled it also reports Tiny and Enigma. *VIRx* reports two different viruses (Syslock and Datacrime) which are detected with the search patterns published in *VB*. *F-PROT* reports 'Possibly a new variant of Plaice' while *FindViru* from *S&S International* does not report anything at all.

There is an obvious problem with such mis-identification - any attempt by a program to disinfect a file based on a mis-identification will probably result in the original program being irreparably damaged. However - viruses which deliberately include fragments of other viruses are unlikely to spread far - the chances of some scanner detecting them are too high.

### Multiple Patterns

It is still an open question which approach is best - using 'family' search patterns, where the number of variants detected with one pattern is maximised, or adopting the specific approach - using one search pattern for each variant.

Family patterns have one advantage - they increase the chances that any new variants will be detected. On the other hand, specific patterns provide at least minimal identification.

The selection of search patterns in *VB* is not fully consistent - sometimes family patterns are used, but in many cases a new search string is published when new variants appear.

However, those specific signatures are often chosen so they can be combined into one family signature, by using a single wildcard or two. As an example take the signatures published for members of the Parasite family.

```
Parasite     ACB9 0080 F2AE B904 00AC AE75
             EDE2 FA5E 0789 BCF9 008B FE81

Parasite 2   ACB9 0080 F2AE B904 00AC AE75
             EDE2 FA5E 0789 7C41 908B FE83

Parasite 2B  ACB9 0080 F2AE B904 00AC AE75
             EDE2 FA5E 0789 7C42 908B FE83
```

These three search patterns can be combined:

```
Parasite-gen ACB9 0080 F2AE B904 00AC AE75
             EDE2 FA5E 0789 ???? ??8B FE??
```

This single pattern would be expected to have a better chance of detecting new variants of the virus than any of the three original ones, although at the risk of a higher occurrence of false positives. This is particularly risky in the case of search strings for high-level-language viruses, such as the two strings published for the RNA viruses:

```
RNA (1) 1E57 C43E F601 0657 B800 2050 BFFF
        011E 579A B10B B700 BFAE

RNA (2) 1E57 C43E 0C02 0657 B8F0 1C50 BF19
        021E 579A E50B BA00 E8D3
```

These strings were deliberately chosen so that they include absolute address references, in order to reduce the chances of a false positive in the case of a different program compiled with the same compiler. Combining the strings is not advisable, as the chances of getting false positives is too high.

### Avoiding Heuristics

Although heuristic (rule-based) scanners have not gained any significant popularity, virus authors have started to modify their viruses to bypass the current generation of analysis tools. The Creeper virus is an example. It is a resident virus, which intercepts the execution of other programs and like other similar viruses it intercepts INT 21H and waits until function 4B00H is executed. Instead of the usual infection methods shown below:

```
cmp ax,4b00h          cmp ah,4bh
jz Infect      or     jz infect
```

the virus uses a more complicated routine:

```
push ax
sub ax,4b00h
jz infect
pop ax
```

Such convoluted code is an obvious attempt to avoid detection by heuristic analysis tools which look for obvious programming indicative of virus behaviour.

### Will the Real Bob Please Stand Up...

For some unknown reason, several viruses have appeared recently, containing the name 'Bob', which is causing some confusion.

The first 'Bob' was included in a collection provided by *Microcom*. The sample was named BOB.COM, but structurally it is an EXE file. In fact this file is the first generation sample of a 561 byte variant of the Vienna virus - it simply needs to be run through *EXE2BIN* to be converted into a working virus. As this virus makes no reference to any 'Bob' whatsoever, it should certainly not be named 'Bob', but rather just Vienna-561 or similar.

The second 'Bob' is in the *AVPD* collection. It is closely related to the Freew-692 virus, but is 718 bytes long, so a more suitable name might be Freew-718. Why this virus has been named 'Bob' or 'Robert B.' is not known.

The third 'Bob' is a member of the Phalcon family, which also includes the Ministry virus. Here the reason for the name 'Bob' is obvious, as the virus includes the following text:

```
Bob Ross lives!
Bob Ross is watching!
Maybe he lives here...
What a happy little cloud!
Maybe he has a neighbour right here...
You can make up stories as you go along.
```

No matter who Bob Ross is, it might not be advisable to name a virus after a (possibly) real person, and perhaps it would have been better to select a different name for this virus, such as 'Cloud'.

### A Fishy Problem

The Chinese Fish virus search pattern has been reported to cause numerous false alarms when used in conjunction with IBM's *VIRSCAN* program operating in its defaults 'mutants' mode. The following pattern should be used instead:

```
Chinese  Fish  8A2E  B500  2E8A  0EB6  00B6  002E
               8A16  B700  CD13  5973  02E2  DDE9
```

Note that *VB* patterns should only be used with *VIRSCAN* if its 'no-mutants' option is invoked.

### Conflicts Between Anti-Virus Programs

Most well implemented anti-virus programs keep any virus patterns encrypted both on disk and in memory. This is done simply to avoid false alarms - if a virus scanner from a different company is run, it should not detect any viral fragments left, for example, in the buffers when the program exits. Encryption for this purpose need not be complex - at least two companies use the simple method of reversing the byte order of their search strings.

Unfortunately, not all manufacturers pay attention to this - the *Central Point Anti-Virus* program, for instance, is notorious for causing false alarms when run in conjunction with other scanners (see p.13).

*CPAV* contains a plaintext Flip detection string which is picked up by the following commonly used pattern:

```
0E BB ?? ?? 1F B9 ?? ?? B2 ?? 81 C1 ?? ?? EB ??
```

Scanners which use this pattern to detect the Flip virus thus conflict with *CPAV*. The string, which is from the decryption routine of the virus, is widely used in detection software as it offers the only stable pattern that can be found in all generations of the virus.

The *Central Point Anti-Virus* manual (p.97) advises disabling antivirus software other than that supplied with *CPAV*. With PC Support staff now using a multitude of scanners from a variety of sources, this advice is inappropriate.

If you must use a 'messy' scanner which does not clean up its working area, we suggest that each and every scan be conducted after a cold system reboot. This practice will avert false positives caused by those scanners which leave detection patterns in memory after completing their search.

### OLE Report Postponed

A report on the implications of Obect Linking and Embedding under *Windows 3.1* has been postponed until next month. *VB* is indebted to staff at *Microsoft UK* for their assistance in the preparation of the report.

### Virus Prevalence Table - April 1992

Incidents reported to *VB* in the UK during April 1992.

| Virus | Incidents | Total Reports |
|---|---|---|
| Form | 11 | 22% |
| New Zealand II | 9 | 18% |
| Spanish Telecom | 5 | 10% |
| Dir II | 3 | 6% |
| 1575 | 3 | 6% |
| NoInt | 3 | 6% |
| Eddie 2 | 3 | 6% |
| Cascade | 3 | 6% |
| Joshi | 2 | 4% |
| Datalock | 2 | 4% |
| Tequila | 2 | 4% |
| Murphy | 2 | 4% |
| Vacsina | 1 | 2% |
| Liberty | 1 | 2% |
| Total | 50 | 100% |

# IBM PC VIRUSES (UPDATE)

Updates and amendments to the *Virus Bulletin Table of Known IBM PC* Viruses as of 22nd May 1992. Entries consist 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 using the 'search' routine of a disk utility or preferably using a dedicated scanner which contains an 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

**Seen Viruses**

**5792** (temporary name) - EN: This virus is similar to the RNA2 and Halloween viruses, written in some high-level language (C or Pascal) and adds 5792 bytes in front of infected files.

```
5792            8DBE 00FF 1657 8DBE 5CE8 1657 B8A0 1650 8DBE FCFE 1657 9A25
```

**16850** (temporary name) - PN: This large (16850 byte) companion virus seems to be written in Turbo Pascal. Because of the high chance of false positives, it is recommended that search patterns should not be used to detect it. To get rid of the virus, simply remove all hidden 16850 byte COM files corresponding to EXE files in the same directory.

**BNB**, Beast-N-Black - CN: This 429 byte virus is a Vienna variant. It contains the text 'Beware the Beast-N-Black'.

```
BNB             FC8B F283 C619 BF00 01B9 0300 F3A4 8BF2 B824 35CD 2106 53B8
```

**Black Jec-4B**, 6B, 8B - CN: 252, 281 and 363 bytes long. Very similar to the variants reported as Bljec-4,6 and 8 and functionally identical. Detected with the Black Jec (Bljec) pattern.

**Black Jec-Digital F/X** - CN: This 440 byte variant is extremely badly written. It starts with a block of text, which will totally crash on some types of hardware. However, the virus may work on some '386 machines. Detected with the Black Jec (Bljec) pattern.

**Cannabis** - DR: A Dutch boot sector virus, which contains the text 'Hey man, I don't wanna work. I'm too stoned right now.' The virus is very badly written and barely qualifies as a real virus.

```
Cannabis        B810 008E D8A1 1303 4848 A313 031F B106 D3E0 2DC0 078E C0B9
```

**Close** - ER: This 656 byte virus may damage either C:\IO.SYS or C:\IBMBIO.COM, making the hard disk unbootable.

```
Close           FE0F 1F83 2C31 1E8B CE36 FE07 0726 836C FF31 268E 44FF 33F6
```

**Cookie** - CEN: This virus is not related to the Cookie variants of the Japanese Christmas and Syslock families, but it is a large virus, compiled with one of the Borland compilers. As the name indicates, the virus demands a cookie, but has not yet been analysed, because of its size. Two variants are known, 7360 and 7392 bytes.

```
Cookie-7392     BFD6 3E1E 57BF 4820 1E57 B8E0 1C50 BF5A 3F1E 579A 180B C000
Cookie-7360     BFE2 3E1E 57BF 4820 1E57 B8C0 1C50 BF66 3F1E 579A 8209 D100
```

**Cossiga** - EN: This is a family of two viruses, a 883 byte version, which is clearly older and more primitive, and a 1361 byte variant which contains the string 'FRIENDS OF MAIS and CLAUDIA SAHIFFER'. Awaiting analysis.

```
Cossiga         8BC1 83E1 0FBB 1000 2BD9 53F8 8B55 1C03 C383 D200 B910 00F7
Friends         5158 83E1 0FBB 1000 2BD9 53F8 8B55 1C03 C383 D200 B910 00F7
```

**Creeper-252** - CR: Similar to the variant reported earlier.

```
Creeper-252     C6FE C60E 07CD 2750 2D00 4B74 2558 3DFF 4375 148B 4450 908B
```

**Danish Tiny-Brenda**: This 256 byte virus is similar to the 251 byte version, but the effects are different - when an infected program is run it may occasionally display the text '(C) '92, Stingray/VIPER Luv, Brenda'.

```
Danish-Brenda   8BD7 B902 0090 B43F CD21 813D 0708 74DD 902B D22B C9B8 0242
```

**Dutch Tiny-99** - CN: One of the smallest viruses to infect without overwriting existing files. It does nothing but replicate.

```
Dutch Tiny-99   93B4 3FCD 2180 3C4D 741D B002 E820 0097 B963 00B4 40CD 21B0
```

**Dutch Tiny-124** - CR: Another small virus from the Netherlands, probably written by the same author as the previous one. Rather badly written and crashes on certain types of hardware.

```
Dutch 124       930E 1FB4 3FCD 218B F280 3C4D 741C B002 E8CF FF97 B97C 00B4
```

**Eliza** - CN: This 1193/1194 byte virus works very badly. It damages EXE files, instead of infecting them, and second-generation copies of the virus will usually not work.

```
Eliza           FFE0 5E81 C600 01BF 0001 5951 56AC AAE2 FC5F 5932 C0AA E2FD
```

**EMF** - CN: This 404 byte virus conatins the text 'Screaming Fist', but is quite different from the Screamer virus - perhaps only written by the same author. Not fully analysed.

```
EMF             E810 00B4 408B D583 EA03 B993 01CD 21E8 0100 C3B9 4C01 8DB6
```

**Enemy** - CER: This virus is somewhat difficult to detect, as the length is variable, and it uses a self-modifying encryption routine. The virus includes the text 'I am a stranger in a strange land'. No effects have been found.

```
Enemy           935D 8BF5 56B0 ??B9 AF02 ??2E 3004 46E2 F9C3
```

**Enola** - CER: A 1864 byte virus, probably of Russian origin. Awaiting analysis.

```
Enola           FF74 081F 8ED8 B800 0150 C38C C805 1000 8BD0 2E03 1608 01BE
```

**Europe '92-424** - CR: Three bytes longer than the original variant, but very similar, and detected with the same pattern.

**Flash-Gyorgy** - CER: Like the Brenda and Milana viruses, this variant of the Flash virus seems to be written by a lovesick virus author. In this case the message is 'I LOVE GYÖRGY'.

```
Flash-Gyorgy    1E06 0E1F FCE8 0000 5E8B DE83 C30E B000 FAD5 0A88 07EB 10EA
```

**Freew-718**, Bob - CN: This 718 byte virus seems rather badly written. It overwrites the first 698 bytes of files, storing the overwritten code at the end. The virus activates in January 1993, but the exact effects have not been fully determined.

```
Freew-718       81F9 C907 7206 80FE 0175 0145 B200 BE00 0016 1FB4 47CD 210E
```

**Gotcha-E** - CR: A 607 byte version of the Gotcha virus, which includes fragments from other viruses, for the purpose of misleading virus scanners. It is detected by the *VB* patterns for Syslock and Datacrime (1280), but the following pattern is specific to this virus.

```
Gotcha-E        8BF0 BF00 01B9 0800 F3A4 B800 0150 B8DA DACD 2180 FCA5 7403
```

**Hafenstrasse-1641** - CEN: Just like the 1689 byte variant, this virus 'drops' the Ambulance virus. It is detected with the Hafenstrasse-Kilroy pattern.

**Helloween** - CER: The name of this 1376 byte virus is derived from the string 'HELLOWEEN', which is stored inside it in encrypted form. This virus is totally unrelated to the Halloween virus.

```
Helloween       B440 EB02 B43F E815 0072 022B C1C3 33C9 33D2 B802 42EB 0733
```

**HH&H** - CR: A 4091 byte encrypted virus, which contains the curious string 'HARD HIT & HEAVY HATE the HUMANS !!'. Awaiting analysis.

```
HH&H            50B9 FB0F 8B1E 0101 81C3 1501 8037 ??43 E2FA
```

**Horror** - CER: An encrypted, 2319 byte virus.

```
Horror          8BFE 83C7 0AB9 4E04 2E8A 849D 042E 3005 FEC0 47E2 F8C3
```

**Intruder** - EN: This 1319 byte virus seems to delete infected files occasionally, and infected programs sometimes 'hang', but this seems to be due to sloppy programming, rather than a design feature. Two minor variants are known, A and B, but both are detected with the following pattern.

```
Intruder        5F32 C0AA B001 0AC0 C35F 32C0 C3BA 0600 B41A CD21 BFAF 00BE
```

**Jerusalem-IRA** - CER: What primarily makes this variant different from the standard one is the inclusion of a long list of encrypted names, as well as text referring to the *Irish Republican Army*.

```
Jerusalem-IRA   2638 05E0 F98B D783 C203 B800 4B06 1F0E 07BB 2100 9C2E FF1E
```

**Jerusalem-CNDER** - CER: A minor variant of the 1808/1813 byte standard version, with the self-recognition code changed from 'sURIV' to 'CNDER'. Detected with the Jerusalem-US pattern.

**Jerusalem-Tobacco** - CER: This variant is almost identical to the AntiCad-2900 variant, with little more than a few encrypted text strings changed. It is detected with the AntiCad-2576 pattern.

**Jerusalem-Triple** - CER: A patched minor variant of the 1808/1813 byte standard version, with the self-recognition code changed and a few patches. A sample with the name 'Dragon' appeared which is virtually identical. Detected with the Jerusalem-US pattern.

**Keypress-1744** - CER: Not fully analysed, but does not seem to be significantly different from the other variants.

```
Keypress-1744   3F02 7405 C707 0200 F9F5 1FC3 F606 1801 0174 0D8C C005 1000
```

**Kit** - CER: This virus has one serious 'bug' - it will re-infect the same file repeatedly. It is 2384 bytes long. Inside the virus is the text 'Copyright 1991-1999. KIT VIRUS (version 2.0).' Awaiting analysis.

```
Kit             2EC5 1619 00B8 2425 CD21 071F 5F5E 5A59 5B58 9D2E FF2E 1100
```

**Ko-407**, Dodo-Pig, GIP - CR: Closely related to the Ko-408 virus reported earlier. It contains the text 'GIP'. There is yet another variant, 408 bytes long, which contains the text 'Birdie Hop!'. It is detected with the same search pattern.

```
Ko-407          B802 4233 C9BA FFFF CD21 508B D033 C9B8 0042 CD21 0E1F B43F
```

**Leprosy-Busted** - CN: A primitive, encrypted, overwriting virus.

```
Leprosy-Busted  8B0E 0B02 51E8 0F00 5BB9 3B02 BA00 01B4 40CD 21E8 0100 C3BB
```

**Malaga** - CER: One of the relatively rare multi-partite viruses. It is 2610 bytes long, but in addition to infecting files it will also infect DOS boot sectors on diskettes and hard disks.

```
Malaga          2D04 00A3 1304 B106 D3E0 2DC0 078E C08B F48B FEB9 0001 2EA3
```

**Marauder-560** - CN: This seems to be an older and more primitive variant of the Marauder virus. One significant difference is that the encryption routine is not polymorphic.

```
Marauder-560    0056 5D81 C646 018B FEFC AD33 8619 01AB 49E3 02EB F555 5E5A
```

**Milena** - CER: This 1160 byte virus contains various pieces of code which seem to have been copied from the Dark Avenger virus. The name is derived from the string 'I Love Milana', but the effects are not fully known.

```
Milena          A4A5 1F8B 2606 0033 DB53 FFE0 BA10 00F7 E2C3 558B ECFF 7606
```

**Munich** - CN: An encrypted 2355 byte virus. Awaiting analysis.

```
Munich          0E5A 8EDA 8D36 1900 8BFE B98E 0490 065B 8EC2 AD35 ???? ABE2
```

**Murphy-Tormentor-D** - ER: This 1040 byte variant is closely related to the Tormentor-A and Tormentor-B variants. It is detected with the HIV pattern.

**Nines Complement-776,706** - CR: Two new variants have appeared, where the initial decryption routine has been modified, in order to bypass scanners detecting the original version.

```
Nines  Comp-766  E800 005B BE0E 0003 F3B9 F402 301C 46E2 FB
Nines  Comp-706  E800 005D BE17 0001 EEB9 A502 89F7 8BDD 81EB 0601 AC30 D8AA
```

**NKOTB**, Cover Girl - CN: A 723 byte overwriting virus, where most of the virus body contains a silly message.

```
NKOTB           BA00 01CD 21B4 3ECD 219F B908 00D3 C82B C183 DB02 5AB8 0157
```

**Plaice** - CR: 1129 bytes. Awaiting analysis. A variant of this virus exists, which has not yet been named, but the sample circulating in the anti-virus community is named 1720C.COM. This is a variable-length, polymorphic variant, with a base length of 1701 bytes. It does not work properly on certain types of hardware. No search string is possible for this variant.

```
Plaice          0001 5033 C033 DB33 C933 D233 F633 FF33 ED0E 0E07 1FC7 0600
```

**Plovdiv 1.3B** - CR: This virus is 1000 bytes long, but is only slightly different from the 1.3 variant.

```
Plovdiv 1.3B    80E2 1F80 FA1E 7506 2681 6F1D E803 075A 5B9D CA02 00B4 1A5A
```

**Quiet** - CR: 2048 bytes. Awaiting analysis.

```
Quiet           A12C 008E C0BB FFFF 4326 803F 0075 F926 807F 0100 75F2 83C3
```

**RNA** - CEN: Like many other large viruses, this one is written in some high-level language and adds itself in front of the files it infects. Version 1 is 7296 bytes long, whereas version 2 is 7408 bytes long.

```
RNA (1)         1E57 C43E F601 0657 B800 2050 BFFF 011E 579A B10B B700 BFAE
RNA (2)         1E57 C43E 0C02 0657 B8F0 1C50 BF19 021E 579A E50B BA00 E8D3
```

**Scion**, Doomsday One, Null Set - CN: This virus is 733 bytes long and contains potentially destructive code (INT 26H calls). The virus is encrypted, and as the decryption routine is very short, only a partial search string is possible.

```
Scion           A003 01B9 CE02 BE?? ??8B D928 00E2 FA
```

**Screamer II**, Screaming Fist II - CER: This variant of screamer immediately infects COMMAND.COM as it goes memory-resident. Program files are infected as they are executed or opened for any reason (including getting/setting file attributes). The virus is prevalent in the United States. Probably written by the same person as wrote the Screamer (Screaming Fist) virus, but more 'advanced' - it is now 838 bytes long and includes limited polymorphic ability, but can be detected with a wildcard string.

```
Screamer II     5D55 ???? 2E8A 8640 032E 8A96 4103 B91B 03F6 ??30 ??2E 30??
```

**SHHS** - CN: A 585 byte overwriting virus. Extremely unlikely to spread, but contains code to trash the hard disk.

```
SHHS            01C3 BB3E 01A0 0601 0AC0 740B 3007 4302 C781 FB49 037E F5C3
```

**Shirley-Vivaldi** - ER: This is a variant of the Shirley virus, with the same infective length as the original, 4096 bytes. Awaiting analysis.

```
Vivaldi         B887 4BCD 213D 6366 7566 2EA1 0E0E 8CDB 01D8 0510 008E D031
```

**Stupid-Profesor** - CR: Almost identical to the SADAM variant, but the text string has been changed to 'The Profesor is in town again'. Detected with the SADAM (Saddam) pattern.

**Suriv 1 Anti-D** - CR: This variant of the Suriv 1 or 'April 1st' virus was discovered in Argentina. It is 945 bytes long and interferes with the 'D' key on the keyboard.

```
Anti-D          0E1F C606 4801 00B4 2ACD 2181 F9C4 0772 0C81 FA11 08EB 0690
```

**Suriv 1-Xuxa** - CR: Yet another Suriv 1 variant from Argentina. It is reported to play music between 5 PM and 6 PM. Infective length is 1413 bytes.

```
Suriv 1-Xuxa    0E1F B42A CD21 81F9 C407 720D 81FA 0208 7402 7205 C606 1E02
```

**SVS** - CR: This virus has been reported elsewhere as Terminator, but that name should be avoided, as it conflicts with the other Terminator viruses. It is 526 bytes long and activates on December 25th, when it displays the message 'TERMINATOR 1991. Made by SVS-009'

```
SVS             B104 D3EB 83C3 11B4 4ACD 21D3 E34B 4B8B E3B8 2135 CD21 2E89
```

**Tack** - CN: A simple 449 byte virus, which may display the message 'Hello, I am virus'. The virus appends itself to the end of infected files, and overwrites the first six bytes, but only restores the first five, which may result in unpredictable behaviour of infected files.

```
Tack            5850 0500 01A3 3C02 C706 3E02 FFE0 C606 4002 23B4 408B 1E33
```

**Trivial-30D** - CN: Yet another attempt to create the smallest overwriting virus. The virus has no side effects.

```
Trivial-30D     CD21 BA9E 00B8 013D CD21 938B D6B1 1EB4 40CD 21C3 2A2E 2A00
```

**Trivial-Hastings** - CN: This overwriting virus is 200 bytes long, but most of that code is taken up by a long text message. The virus does nothing but replicate.

```
Hastings        B802 3DBA F001 CD21 720C 8BD8 B440 B9C8 00BA 0001 CD21 CD20
```

**Vienna-637** - CN: Very similar to the original version, and detected with the Vienna-1 pattern.

**Vienna-712** - CN: This variant seems most closely related to the Dr Q. variant, and uses limited encryption. It is detected with the Vienna-4 and Dr Q. patterns.

**Vienna-Parasite-2** - CN: This virus is 901 bytes long and is closely related to the Parasite and Parasite-2B variants.

```
Parasite 2      ACB9 0080 F2AE B904 00AC AE75 EDE2 FA5E 0789 7C41 908B FE83
```

**Vienna-Betaboys** - CN: This 679 byte variant was written in Sweden, or possibly in Finland. It activates in February of any year, trashing the beginning of drives C, D and E.

```
Betaboys        90AC B900 80F2 AEB9 0400 ACAE 75EA E2FA 5E07 897C 2490 8BFE
```

**Vienna-Violator-B2** - CN: This 969 byte variant is not new and is not expected to become a serious threat, as it only works properly for a single generation - after that copies seem to be corrupted.

```
Violator-B2     90AC B900 80F2 AEB9 0400 ACAE 75ED E2FA 5E07 897C 4E90 8BFE
```

**Violetta-1024** - CN: Probably just an earlier variant of the Violetta virus. This variant has also been reported as 'Thimble'.

```
Violetta-1024   B425 B0FF 061F 89DA CD21 0E1F B425 B021 BA00 03CD FFB4 31BA
```

**VVF 3.4** - CR: This Russian virus only works on some machines, but crashes on certain types of hardware, such as the IBM XT. Awaiting disassembly.

```
VVF 3.4         7606 81C3 0001 8BF3 FCF3 A41E BB00 0153 CB8C D848 8ED8 8B1E
```

**Yankee-1905/1909** - CER: Also known as the '83', this variant is slightly unusual in that EXE files grow by 1905 bytes, but the virus adds 1909 bytes to COM files. Detected with the Yankee pattern.

**Yankee-Login** - CER: This 3045 byte variant of the Yankee Doodle virus has been reported to operate as a password 'snatcher' on a network, and to cause irreversible damage to data. It does not seem to work on certain types of hardware, including XTs with monochrome displays. At least four minor variants have been reported, but they are virtually identical, and have the same length.

```
Yankee-Login    B440 EB02 B43F E809 0072 023B C1C3 32C0 B442 2E8B 1E3A 00CD
```

# HEADLINES

## Ludwig Shelters Under First Amendment

Mark Ludwig, author of *The Little Black Book of Computer Viruses* (see *VB*, March 1992, pp 17-18) is pleading the *First Amendment* in order to justify the publication of the book. In a communication on *Compuserve* he stated 'I have consulted with a lawyer regarding the legality of writing/publishing viruses, and have found that among the legal community, it seems unanimous that writing and publishing viruses, per se, is TOTALLY protected by the First Amendment.'

For anyone unlucky enough to be hit by one of his viruses, Ludwig warns: 'I don't have enough money to make fining me or suing me profitable, but I'd bet the press coverage of such a move would do me a lot of good, and you'd end up looking like the evil tyrant trying to destroy every American's freedom and usher in the dark new world order.'

His warnings are not just aimed at his victims seeking criminal or civil redress. While he does not have enough money to face legal proceedings himself, he *is* prepared to sue anyone who distributes any part of his viruses (including, by definition, search patterns). 'Have you considered the fact that the software in my book is copyrighted?', he asks. 'If someone uses it to create a malignant virus in any country, or even allows what is in the book to go off replicating, he is violating my copyright and using it illegally. (Or if any anti-viral types start distributing it without paying royalties, they are breaking the law too.) I would be fully justified in initiating the prosecution in such cases.'

Having pleaded poverty, Mr Ludwig's veiled threat to sue all the major players in the anti-virus industry appears particularly hollow. *VB* has already published patterns lifted from the book: yet the arrival of a writ does not appear imminent!

Early in May, a message was placed on *Compuserve's Virus Forum* from a victim who claimed that a Ludwig virus had attempted to corrupt his File Allocation Table. Ludwig's reply showed little remorse: 'To suggest that the viruses in my book TRY to trash your FAT table suggests intent, and is out-and-out libelous.' But then he half-admits intent by saying that the Kilroy virus isn't very 'smart' and will cause corruption if 'your hard disk isn't partitioned in the normal way or isn't loaded with DOS....'. Ludwig continues: 'I'll be happy to see to it that the code gets fixed and updated in the next printing.' The victim's reply is understandable: 'Excuse me, but if you think that my note is libelous, take me to court you jerk!'

The book has not surfaced in the United Kingdom (it is not included in *Books In Print*, the book trade's bible) despite Ludwig's claim of high sales in the UK. It recently transpired that Ludwig, having had his *meisterwerk* turned down by a succession of publishers, actually published his book himself - i.e. Mark A. Ludwig is *American Eagle Press*.

### 40Hex - The Virus Writer's Magazine

An electronic magazine entitled *40Hex* is circulating on underground bulletin boards. The magazine seems to be the virus writers' equivalent of *VB*. Edition 6 contains articles on subverting memory-resident anti-virus programs, observing scanner strings in memory, hacking virus code to make it undetectable and source code for the DIR II virus. *40Hex* is 'published' by the North American Phalcon/Skism group which operates the *Digital Warfare* BBS. Contributors include such luminaries as Dark Angel, Decimator, Garbageheap, Hellraiser and Night Crawler. Issue 6 announces a virus writing contest, the closing date for which is July 4th. The magazine's title is derived from the fact that 40 Hex produces the ASCII @ character beloved by e-mailers everywhere.

```
                Virus Contest!
              'The Spammies(tm)'
           Deadline: July 4th, 1992

Rules and Regulations:

1) All submissions must be original source code.
(no hacks)

2) Only one submission is allowed per programmer,
plus one group project.

3) All viruses must be recieved (sic) before July
4th, 1992.

4) Viruses must be accompanied by a complete
entry form. (see below)

5) The original, compilable, commented source
MUST be included, along with an installer
program, or a dropper, in the case of boot block
viruses.

6) Entries must include a location where the
author may be contacted, such as an email address
or BBS.

7) Personnel or persons related to personnel of
PHALCON/SKISM are not eligable (sic).

8) The source must compile without error under
Tasm or Masm (please specify what assembler and
version you used, along with the necessary
command line switches). If we cannot compile your
virus, it will be disqualified.

9) All entries recieve (sic) a free subscription to
40hex. (hehehehe)

10) The entry must be uploaded privately to the
sysop, stating that it is a contest entry.

11) The viruses must not be detectable by the
current version (as of July 4th) of any known
virus scanner.

12) Viruses will be judged by our 'panel of
experts' in three (sic) catagories (sic).

6.1) Stealth

6.2) Size

6.3) Reproductivity

6.4) Performance
```

*40Hex* announces its illiterate and innumerate competition - it is no wonder that viruses are so bug-ridden!

# SHAREWARE

***Fridrik Skulason***

## A Developer's Perspective

The shareware concept is simple - software is distributed freely and users are encouraged to share copies with friends. Anybody who decides to use the program is expected to register the product and pay for it. This 'try before you buy' approach sometimes works surprisingly well, but most shareware authors have been disappointed by meagre returns.

Why is software distributed as shareware in the first place? In some cases this is because the software failed as a full-blown commercial product. In other cases the producer did not have the necessary resources to produce, market and support the program to the extent that shrink-wrap packages require. Many shareware programs are the work of hobbyists or part-time devotees and the shareware route was adopted due to its extremely low distribution and marketing costs.

### Small is Beautiful - Sometimes

Shareware outfits are generally small - sometimes just one person working part time, and although small companies don't have the same resources as bigger companies, they do have one distinct advantage - they can move rapidly and undertake drastic design changes with no committee meetings to slow things down. This is particularly relevant with regard to anti-virus software whereby an updated scanner can be distributed to the major archives in a matter of hours. However, the relatively small size of most shareware operations has its drawbacks; technical support, for instance, may not be as readily available from a shareware developer as from a large commercial software house. That said, if a shareware user *does* contact its developer he will receive unparalleled assistance - no-one will know the product more intimately!

To succeed, shareware must appeal to a significant proportion of the user community - otherwise it will not be distributed. Anti-virus software certainly meets this condition. Apart from games, anti-virus software is probably the most frequently used type of shareware.

Successful shareware must be as good as competing shrink-wrap packages. Whether this holds true for the current shareware anti-virus programs is an open question - there are a few excellent shareware packages available, but also some feeble ones. (It must be noted that a few of the commercial products are equally feeble.) From a purely technical point of view there does not seem to be a significant difference between the best shareware and shrink-wrap packages. The major differences are in presentation, documentation, and to a certain degree in the level of support provided. The most widely used anti-virus shareware packages for the PC include *SCAN* and *CLEAN* (*McAfee Associates*), my own *F-PROT* (*Frisk Software*), Ross Greenberg's *Flushot+* (*Software Concepts Design*) and Rich Levin's *CHECKUP*. There are a number of shareware Macintosh offerings including *Disinfectant* (John Norstad, *Northwestern University*, Illinois), *Gatekeeper* (Chris Johnson) and *Virus Detective* (Jeff Schulman) which are available from numerous archives including *Compuserve*, *AppleLink* and *MacNet*.

### Corporate Obstinacy and Other Obstacles

Shareware programs have a significant share of the anti-virus market, despite certain difficulties. One problem is that some organisations have a very strict 'no PD - no shareware' policy, or that they simply don't take the shareware product seriously. The shareware developer fights a constant battle against entrenched attitudes and scepticism. Remarks such as 'You're trying to tell me that this shareware product is as good as a package costing twenty times as much?' are all too common.

Another problem shareware authors face is getting users actually to *register* the software. In certain parts of the world, shareware is considered synonymous with 'freeware', and even in countries like the USA and Canada where the shareware concept is well accepted, only a fraction of users of shareware products actually pay for them. Several 'tricks' are used to encourage people to register - they may be offered a 'full version' of the package, with the shareware version just being a bait. Another approach is even more effective; anti-virus software is special in one way - virus-specific programs, such as scanners become obsolete in a very short time. Regular updates thus provide a convincing reason to register.

### The Major Bulletin Boards

Shareware products are generally distributed on BBSs and through on-line services such as *Compuserve*; (by far the largest BBS worldwide - USA; 614-457-0802, UK registration; 0800-289-378), *BIX* (operated by *Byte* magazine; 603-924-7681), *GEnie* (*General Electric*; 800-638-9636) and *Delphi* (*General Videotex Corporation*; 617-491-3393).

These distribution channels have certain advantages - they provide fairly fast, worldwide distribution at a very low cost. This makes frequent updates possible, but these channels have one serious drawback. Occasionally shareware programs have been Trojanised and modified versions released. The most widely available shareware anti-virus program (*McAfee's SCAN*) has been Trojanised several times and the modified files uploaded to BBSs.

If shareware is obtained directly from the developer, or from a reliable source, this problem of tampering and subversion does not arise; assuming the program source is secure, shareware programs are no riskier than shrink-wrap packages - they are just marketed in a different way.

# ✉ LETTERS

Dear Ed,

I noticed the technical notes article entitled 'Bombed Out' (*VB*, May 1992) and thought I might make a few comments.

A major reason small companies use outside programmers is that they cannot afford the tools required for program development, the expertise that goes with it and the problems associated with more on-site personnel. They often prefer that programmers work at home and on their own time. In the US, this is a requirement for the legal distinction between an employee and a consultant for tax and benefit purposes. The suggestion that contractors work on-site thus has problems.

Many small companies which use outside programming talent don't know how to do systems maintenance that well. Most contractors thus need *unrestricted* access. Many outside programmers maintain source rights to their code and will *only* provide binary executables to the customer. This reduces costs by allowing the programmer to reuse code between jobs.

If you can afford personnel to review the software as you suggest, you can afford personnel to write it! Really sound change control takes two change control experts for each source programmer (according to those who do it for a living).

As for soliciting assistance from the software manufacturer, did you ever try to get *Microsoft* (or any other similar company) to write a custom program? This costs more than the annual budget in most small companies! *AT+T* will write Unix programs for $200 per hour (these prices are several years old - it's probably $250 by now). At this rate, I could hire five full-time 20-year veteran systems programmers with expertise in my business area, provide them with their own '486 systems in a network, and have money to spare for secretarial help. How many small businesses can afford that?

I think we should have a bonding system. When an unscrupulous programmer is caught, the company gets restitution through the bond and the programmer is unable to work again because no bondperson in their right mind would grant a bond to someone who has done this. The insurance companies should refuse to pay dividends for losses resulting from the use of non-bonded programmers. Companies would thus be forced to use bonded programmers and the punishment for malpractice would be long term and severe. Don't get me wrong - I don't want a certification system nor a mechanism that introduces significant barriers to programmers, but measures to secure the rights of clients and programmers and which punish the malicious programmer. I should also note that criminal and civil law can be used to affect restitution.

Sincerely yours,

*Dr Frederick B Cohen*
*President, ASP*

Sir,

I read your 'Head-Rolls' column in the May 1992 *Virus Bulletin* and would like to offer a couple of comments that clarify some incorrect information provided in that article:

1. The baseline library used in the February certification test came from Joe Wells at *Certus*, not John McAfee. Subsequent contributions were made by John McAfee, Patti Hoffman and Joe Wells. All *AVPD* members have had full access to this library from its inception in December 1991 and all have an equal opportunity to contribute and maintain the collection. Anyone 'sitting on the sidelines' was doing so by their choice. To refer to this library as the McAfee Suite represents gross pandering to your readers rather than honest, objective reporting of the facts.

2. Patti Hoffman maintains a library which is totally independent of the official *AVPD* library. She uses this separate library in her *VSUM* certification process. She has never conducted official tests using the *AVPD* library. Until March 1992, David Stang conducted all tests which used the library.

3. Vendors have expressed concern about past *NCSA* testing and the lack of notification of upcoming tests. A formal set of guidelines was published when I assumed responsibility for testing. This guideline was reviewed and accepted by the *AVPD* advisory board in Portland on April 24th.

4. David Stang must have been misquoted in the *Newsday* article. Otherwise, David is saying that he allowed the test to be unduly influenced by John McAfee. I have too much respect for David's integrity to believe that.

There is slim justification for printing the inaccuracies represented in this article. I hope you will exercise more restraint in the future before stooping to the reporting of such inaccurate and inflammatory material.

Sincerely,

*Robert C. Bales*
*NCSA, Pennsylvania, USA*

[*Mr Quittner's Newsday article has been widely circulated in the industry and has been the subject of much comment; his report about the rift between Mr McAfee and Dr Stang over NCSA testing merited an echo in VB. Mr Quittner teaches at the Columbia School of Journalism and was one of five staff reporters to be awarded the Pulitzer prize for investigative reporting this year. If his report was incorrect I am quite sure that legal proceedings would have commenced very quickly. By Mr Bales' own admission, John McAfee was involved in the composition of the test-set. Moreover, Patricia Hoffman, while not conducting AVPD tests, was removed from her position as official AVPD librarian at the Portland AVPD meeting on April 6th 1992. The VB report reflected feelings of unease among many manufacturers surrounding recent product evaluations conducted by NCSA. However, I agree that certain points of detail in the report were inaccurate. Ed.*]

Dear Ed,

Well of all the cheek! Imagine actually going out and doing a real-life survey of support provided by software companies. Heavens, where have all the sacred cows gone? And what a sample. One phone call, or was it two? Makes it easy to calculate the average company response time I suppose. Of course to be really representative you do it in the wee hours, when getting first aid for mortal wounds can be impossible. Then you call our distributor when the only phone numbers on our product are for Australia and the USA. You wouldn't have even had to wake us up!

Seriously though, we think your initiative has thrown up some real challenges. Support is of paramount importance to most users of anti-virus software, particularly corporate users. Irrespective that most quality software would allow a minimally competent user to remove this particular virus, it is a fact that many users want to talk to someone. It falls to us as developers and suppliers to ensure that this service is costed into our product and then properly delivered.

We accept that on this occasion our response was not the best and we apologise for that. We have taken steps to improve our technical response capability, particularly with after-hours backup for the UK. This will be enhanced with the establishment of a *Leprechaun Software* UK office.

Incidentally, it is our policy to ask for a serial number. Our customers appreciate that they pay us to look after them, not the countless pirates and free-loaders. Obtaining the serial number also establishes that the user is in fact using our product and which version is in use. If the user is confused as to whose product he or she has (it happens) our instructions will be misleading and possibly disastrous. Knowing the version number helps in quickly locating problems.

Again, congratulations on raising a real issue and keep up the good work. Next time though, quote your serial number, or at least think up a better excuse!

*Jack Kenyon*
*Leprechaun Software*

Sir,

I was dismayed by the inaccuracies in your review of our service.

Your researcher called to ask how to use '*Virex-PC* version 1.8' to disinfect a New Zealand 2/Stoned drive. However, there has never been such a version of our product. *Microcom* does, however, release *VIRx*, a freeware, detect-only version of our scanner that is updated monthly. Since *VIRx* has no disinfection capabilities, it is routine for our technicians to recommend that customers who have detected a virus purchase our commercial package to repair the infection!

As to the report that the researcher was on hold for seven minutes, this is virtually impossible given the design of our phone system and the way we handle support calls. *Microcom* does not use a voicemail system and, as a result, any incoming call is taken by one of our receptionists. Once the call has been answered, it is either passed to a technician or, if no technician is immediately available, a message is taken so that the call can be returned. If the call is passed to a technician, it can only stay on hold for 30 seconds before it rings back to the receptionist. It is then immediately answered and a message is taken for a call-back.

Finally, I would like to point out that the documentation that accompanies the commercial product references the phone number for the office which supports *Virex* for the PC (the one given in your original article) and no other.

Sincerely,

*Jon K. Ward*
*Microcom, Inc.*

[*The use of the VIRx scanner in this test was a mistake - apologies to Microcom for this oversight. Microcom has consistently sent copies of VIRx to VB, but only beta-test versions of Virex-PC which has caused some confusion. Regarding the second point, the researcher maintains that she was kept on hold for seven minutes; a fact recorded in the log written during the survey*.]

Dear Editor,

I wholeheartedly agree with Richard Jacobs' criticisms of Mark Ludwig's irresponsible work, *The Little Black Book of Computer Viruses* (*VB*, March 1992). However, I must object to the inaccuracies and prejudices Mr Jacobs expresses regarding the laws of the United States.

Mr Jacobs suggests that the Mr Ludwig will defend publishing virus source code by 'tak[ing] cover beneath the Fifth Amendment.' If Mr Jacobs is referring to the United States Constitution's guarantees of free speech and a free press, then his reference should be to the First Amendment, not the the Fifth.

Mr Jacobs further states that Mr Ludwig comes 'from a country where gun control is virtually non-existent'. I could just as well say that Mr Jacobs comes from a country where guarantees of free speech and a free press are virtually non-existent. Both statements reek of inaccuracy, subjectivity, irrelevance and baseless prejudice. Mr Jacob's statement has no place in his article or your publication.

Sincerely,

*Paul E Milligan,*
*Columbia, Maryland, USA*

[*Mr Milligan is right to say that the United Kingdom has no guarantees of a free press or free speech; the UK has no written constitution. Interestingly, Mr Ludwig is quoting the First Amendment as a defence for his activities! In the light of recent events in Los Angeles I find it hard to consider Richard Jacob's 'baseless prejudice' on US gun control as wholly inaccurate or irrelevant. Ed*.]

# SCANNER UPDATE

***Mark Hamilton***

## Three Essential Criteria

In the following test, scanners were assessed for:

➤ their ability to detect viruses known to be in the wild.

➤ their ability to detect selected polymorphic strains reliably.

➤ their concordance with each other.

Any well-maintained scanner should achieve a perfect rating in these tests: with the exception of a few viruses in the wild (Tequila, Michelangelo, Maltese Amoeba etc.), the virus test-sets have remained unaltered since October 1991. Seventeen packages from sixteen suppliers were tested. With two possible exceptions, all scanners were the latest versions.

The In The Wild test set comprised a total of 105 infected files and 11 genuine boot sector virus infections. Note that some encrypting viruses such as Tequila and Spanish Telecom have now been transferred into the In The Wild test-set. The Polymorphic set consisted of a total of 150 file infections. Full details of the test set are published on page 16. The detection percentages reported refer to the percentage of infected files detected, not the percentage of viruses detected.

The Concordance test ascertains whether each package can coexist with all the others or whether one of its constituents causes another virus package to detect a false positive.

The three tests featured this month were limited in their objectives and this review is far from comprehensive. However, it does provide an insight into the relative capabilities of the scanners on trial. The In The Wild test identifies those products which provide an adequate *baseline* of defence. The polymorphic test (previously called the 'stamina test') singles out the scanners which contend successfully with variably encrypting viruses and helps to determine the level of commitment and skill shown by each developer in dealing with the growing number of these specimens. The concordance test reveals which products are likely to fall foul of each other, thus causing time-wasting false positives.

### VISCAN Version 3.27

| | |
|---|---|
| In The Wild | 100% |
| Polymorphic | 100% |
| Concordance | Passed |

This product continues its high standard of virus detection.

### Central Point Anti-Virus for DOS Version 1.2

| | |
|---|---|
| In The Wild | 99% |
| Polymorphic | 60% |
| Concordance | Failed |

*Central Point* has improved its detection capabilities of common viruses but, in all other respects, its performance continues to be relatively poor. It missed an infection of PcVrsDs and all the Bulgarian polymorphic viruses (Evil, Phoenix and Proud).

Since its launch, *Central Point's* product has contained a recognisable pattern for Flip within its memory-resident modules. This release continues to cause false positives in other scanners. Five peer products detect the Flip virus in *Central Point's VWATCH* and *VSAFE* files. The five concerned are *Total Control's VIS, Harry Thijsen's HTScan, IBM's Virscan, Leprechaun's VBuster* and *RG Software's VI-Spy*. It is time *Central Point* solved this problem.

### CPAV for Windows Version 1.0

| | |
|---|---|
| In The Wild | 99% |
| Polymorphic | 60% |
| Concordance | Failed |

*Central Point* is marketing its *Windows* product separately and it was tested to ascertain whether there are significant differences between it and the DOS release. There are no differences in its virus-specific detection capability.

The same five peer products found Flip in its *VSAFE* and *VWATCH* files.

### TBScan - Version 3.3

| | |
|---|---|
| In The Wild | 96% |
| Polymorphic | 93% |
| Concordance | Passed |

A number of changes have been made to this shareware product since *VB's* last review. The most significant new feature is 'Algorithmic Virus Recognition' (AVR) to detect polymorphic viruses. Two AVR modules were included for Maltese Amoeba and the Washburn V2Pn viruses. The Washburn AVR is not completely effective as it missed 20% of the V2P6 infections. *TBScan* failed to detect Dir-II, Spanz and three of the four Spanish Telecom 2 infections; otherwise its performance has improved.

## UTSCAN - Version 19.04

| | |
|---|---|
| In The Wild | 44% |
| Polymorphic | 13% |
| Concordance | Passed |

*Virus Bulletin* has not received any updates to this *Fifth Generation Systems* product since it was released in January; presumably its registered end-users are in a similar position.

*UTSCAN* missed 777, Eddie-2, Maltese Amoeba, PcVrsDs, Spanz, one generation of Spanish Telecom 1, Spanish Telecom Boot, all eleven Whales and all forty Flips. The only Polymorphic virus it found was V2P1. A poor performance.

## F-PROT - Version 2.03b

| | |
|---|---|
| In The Wild | 100% |
| Polymorphic | 100% |
| Concordance | Passed |

A perfect score from this consistently reliable software.

## HTScan - Version 1.17

| | |
|---|---|
| In The Wild | 96% |
| Polymorphic | 93% |
| Concordance | Passed |

Developments of this Dutch shareware package exactly mirror those of its compatriot, *TBScan* which accounts for its identical results. It too uses 'AVR' modules but shares *TBScan's* failure reliably to detect V2P6.

## VIRSCAN - Version 2.1.9

| | |
|---|---|
| In The Wild | 100% |
| Polymorphic | 13% |
| Concordance | Passed |

IBM's *High Integrity Computer Laboratory* has improved *VIRSCAN's* detection capabilities; it now detects all viruses known to be at large. However, its performance in detecting polymorphic viruses is poor, reliably finding only V2P1.

| | Standard | Tequila | Whale | Spanish Telecom 1 | Spanish Telecom 2 | Flip | Boot Sectors | Total |
|---|---|---|---|---|---|---|---|---|
| Number of Infections | 40 | 5 | 11 | 5 | 4 | 40 | 11 | 116 |
| Total Control VISCAN | 40 | 5 | 11 | 5 | 4 | 40 | 11 | 116 |
| Central Point (DOS) | 39 | 5 | 11 | 5 | 4 | 40 | 11 | 115 |
| Central Point (Win) | 39 | 5 | 11 | 5 | 4 | 40 | 11 | 115 |
| Certus NOVI | 39 | 5 | 0 | 5 | 4 | 0 | 0 | 50 |
| ESSaSS TBScan | 38 | 5 | 11 | 5 | 1 | 40 | 11 | 111 |
| Fifth Generation | 34 | 0 | 0 | 4 | 3 | 0 | 10 | 51 |
| F-PROT | 40 | 5 | 11 | 5 | 4 | 40 | 11 | 116 |
| HTScan | 38 | 5 | 11 | 5 | 1 | 40 | 11 | 111 |
| IBM VIRSCAN | 40 | 5 | 11 | 5 | 4 | 40 | 11 | 116 |
| Virus Buster | 40 | 5 | 11 | 5 | 3 | 39 | 11 | 114 |
| McAfee SCAN | 40 | 5 | 11 | 5 | 4 | 40 | 11 | 116 |
| Microcom VIRx | 40 | 5 | 10 | 5 | 4 | 40 | 11 | 115 |
| Norton Anti-Virus | 36 | 5 | 11 | 5 | 0 | 40 | 10 | 92 |
| RG Software VI-SPY | 40 | 5 | 11 | 5 | 4 | 40 | 11 | 116 |
| S&S Findvirus | 40 | 5 | 11 | 5 | 4 | 40 | 11 | 116 |
| Sophos SWEEP | 40 | 5 | 11 | 5 | 4 | 40 | 11 | 116 |
| XTree | 37 | 0 | 0 | 5 | 0 | 0 | 9 | 51 |

Test 1. Viruses in the wild. 105 file infections, 11 boot infections - all products should achieve the maximum score of 116.

## Virus Buster - Version 3.86

| | |
|---|---|
| In The Wild | 99% |
| Polymorphic | 99% |
| Concordance | Passed |

*Leprechaun Software's* product improves with each subsequent release. The only minor glitches to its otherwise exemplary scores are that it failed to detect one of the forty Flip progeny and one of the fifty V2P6 infections.

## Microcom VIRx - Version 2.2

| | |
|---|---|
| In The Wild | 99% |
| Polymorphic | 100% |
| Concordance | Passed |

This version was supplied by *Microcom* although it is a free scanner which is distributed to promote their official product *Virex-PC*. Another strong performance from Ross Greenberg and the *Microcom* team. The only infection VIRx failed to detect was one of the eleven Whale generations - since Whale can be detected with thirty standard hex patterns this may be a case of GIGO in the pattern library.

## SCAN - Version 8.4B89

| | |
|---|---|
| In The Wild | 100% |
| Polymorphic | 100% |
| Concordance | Passed |

Another first-class result for *McAfee Associates*.

## Dr Solomon's Anti-Virus Toolkit Version 5.57A

| | |
|---|---|
| In The Wild | 100% |
| Polymorphic | 100% |
| Concordance | Passed |

A flawless result from *S&S International*.

| | 1226 | Evil | Phoenix | Proud | V2P1 | V2P6 | Total |
|---|---|---|---|---|---|---|---|
| Number of Infections | 20 | 20 | 20 | 20 | 20 | 50 | 150 |
| Total Control VISCAN | 20 | 20 | 20 | 20 | 20 | 50 | 150 |
| Central Point (DOS) | 20 | 0 | 0 | 0 | 20 | 50 | 90 |
| Central Point (Win) | 20 | 0 | 0 | 0 | 20 | 50 | 90 |
| Certus NOVI | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| ESSaSS TBScan | 20 | 20 | 20 | 20 | 20 | 40 | 140 |
| Fifth Generation | 0 | 0 | 0 | 0 | 20 | 0 | 20 |
| F-PROT | 20 | 20 | 20 | 20 | 20 | 50 | 150 |
| HTScan | 20 | 20 | 20 | 20 | 20 | 40 | 140 |
| IBM VIRSCAN | 0 | 0 | 0 | 0 | 20 | 0 | 20 |
| Virus Buster | 20 | 20 | 20 | 20 | 20 | 49 | 149 |
| McAfee SCAN | 20 | 20 | 20 | 20 | 20 | 50 | 150 |
| Microcom VIRx | 20 | 20 | 20 | 20 | 20 | 50 | 150 |
| Norton Anti-Virus | 20 | 20 | 20 | 20 | 20 | 50 | 150 |
| RG Software VI-SPY | 20 | 20 | 20 | 20 | 20 | 50 | 150 |
| S&S Findvirus | 20 | 20 | 20 | 20 | 20 | 50 | 150 |
| Sophos SWEEP | 20 | 20 | 20 | 20 | 20 | 50 | 150 |
| XTree | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Test 2. Encrypting viruses. 150 polymorphic file infections. Note: no Mutation Engine viruses were included in this test.

## Norton Anti-Virus - Version 2.0

| | |
|---|---|
| In The Wild | 92% |
| Polymorphic | 100% |
| Concordance | Passed |

*Symantec* has released a *Windows* version of *NAV* as standard. The product has improved immensely in recent months which may reflect its developer's determination to produce a major league anti-virus tool. It failed to detect PcVrsDs, Slow and Tequila Boot infections but otherwise performed well.

## VI-Spy - Version 8

| | |
|---|---|
| In The Wild | 100% |
| Polymorphic | 100% |
| Concordance | Passed |

*RG Software* will presently release version 9 of *Vi-Spy* which is currently being beta tested.

Version 8 performed flawlessly in these tests.

## Sweep - Version 2.37

| | |
|---|---|
| In The Wild | 100% |
| Polymorphic | 100% |
| Concordance | Passed |

*Sophos'* no-nonsense approach to virus scanning has yielded it with another set of perfect scores. Notice that this product passes the concordance test. A recent comparative review of *SWEEP* (*VNR*, February 1992) criticised a component test file called VIRPATS.BIN for causing false-positives. This file has not appeared in edition of *SWEEP* for some 15 months. Either the review, the scanner, or both were very out-of-date!

## AllSafe/ViruSafe - Version 4.5

| | |
|---|---|
| In The Wild | 44% |
| Polymorphic | 0% |
| Concordance | Passed |

*XTree* and *Fifth Generation* were the only two companies not to supply updated versions of their products. In *Xtree's case*, this has earned them one of the lowest set of scores obtained from these tests. It failed to detect Dir-II, Maltese Amoeba, Spanz, Tequila, Whale, Spanish Telecom 2, Spanish Telecom Boot, Tequila Boot, Flip and all the Polymorphic test-set infections. A very disappointing result.

## NOVI Version 1.0.1

| | |
|---|---|
| In the Wild | 43% |
| Polymorphic | 0% |
| Concordance | Failed |

*NOVI* from American software house *Certus International* is currently being introduced to the UK market. This scanner failed to detect Maltese Amoeba, Whale and Flip. It completely failed the polymorphic detection test. Most alarmingly it did not detect any of the eleven boot sector viruses. A truly gruesome performance.

*NOVI* failed the concordance test because *VIRSCAN* erroneously detected the Mardi Bros virus in the file NOVI.OVL.

### Observations

The majority of these products have either improved or maintained their unblemished record of excellence. *Norton Anti-Virus*, in particular, can be singled out for a striking improvement in its performance. Conversely, *Central Point* really must solve its false positive problem - simply instructing users disable other scanners when using *CPAV* (as its manual advises) is not good enough!

This month's tests featured the largest number of products so far to appear in a *VB* comparative review - the PC user really is spoilt for choice. As a word of caution, advertising copy which claims that product X never needs updating should be disregarded; in the real world they all do.

### The Test Sets

1. In The Wild

Where appropriate one genuine COM and one EXE file infection of: 1575, 777, Cascade (1701), Cascade (1704), Dark Avenger, Dark Avenger 2100, Dir-II, Eddie, Eddie-2, Hallochen, Jerusalem - Friday 13th, Keypress, Liberty 1, Maltese Amoeba, Nomenklatura, Nothing, PcVrsDs, Plastique, Plastique 5.21, Slow, Spanz, Syslock, Vacsina, Vienna (2A), Vienna (2B), Virdem-Generic, Old Yankee 1, Old Yankee 2.

5 generations of Tequila, 11 generations of Whale, 5 generations of Spanish Telecom 1, 4 generations of Spanish Telecom 2, 40 generations of Flip.

The following genuine boot sector infections: Aircop, Brain, Disk Killer, Form, Italian Generic A, Joshi, Korea A, Michelangelo, New Zealand 2, Spanish Telecom, Tequila.

2. Polymorphic

20 generations each of 1226, Evil, Phoenix, Proud, V2P1 and 50 generations of V2P6.

# DETECTION TRENDS

***Jim Bates***

## Preparing For The Inevitable

The discussion concerning the relative merits of virus-specific scanning programs recently took a new and more vehement turn as rumours surfaced that some vendors of scanning software had declared large proportions of their known virus lists 'extinct' or 'rare'. The reason for this was said to be the inability of the software to maintain all of the information in the full list and that by removing details of viruses not confirmed as 'at large', more mileage might be coaxed from their software. The vendors concerned might, with some justification, declare that since only around 10% of known viruses have ever actually been reported 'at large' there is no need to overburden their software. My own feeling is that once a virus is 'known', even if only as a 'lab' virus, users must be protected from its possible arrival. No one can accurately judge what is and is not 'at large' and some research establishments are known to be laughably insecure.

### The Exponential Nightmare

The number of known viruses is said to be doubling roughly every six months. If we assume there were 1,000 samples at the beginning of June 1992 and this doubling continues at that rate, then by June 1997 there will be over one million known viruses. Scanning for even a fraction of this number using present methods will be out of the question and so (the argument goes) virus-specific scanners are a dying breed.

The major advantage of virus-specific scanning is that incoming, unverified software can be examined for known virus code and in a high proportion of cases, infected files can be identified and destroyed before they are introduced into a computing environment. The detection efficiency of the scanner is of prime importance, with speed becoming a factor only when the convenience of the program is considered. However, there are two disadvantages with scanning programs: 1) the constant need for updates as new viruses appear and 2) the increasing need for memory and disk space in which to store and process the relevant search information.

Even without these limitations, scanning software is already encountering difficulties as a direct result of the plethora of new packages finding their way on to the open market.

### Published Patterns

With the increasing numbers of virus specimens, the work necessary to disassemble and analyse them has risen beyond manageable proportions for many vendors and they must rely increasingly upon published recognition patterns (such as those listed in the *Virus Bulletin*).

Some pattern recognition scanners, while quite effective in their detection of known viruses, leave fragments of their search patterns in memory. These fragments are now much more likely to contain published patterns and will therefore result in false positive indications during subsequent use of other scanners (See *Technical Notes*, p.4).

Most scanners encrypt their search patterns, ostensibly to prevent external modification, but also to prevent the pattern file from producing false positive indications (although at least one product does not take this elementary precaution).

### Peaceful Coexistance

As a defence against being targeted by the virus writing fraternity, some scanner developers also encrypt their code in various ways. The choice of encryption methods was never too important as long as it was sound enough to dissuade the hackers. However, while it is sensible for software vendors to avoid using the same encryption algorithms that viruses use, at least one commercial package ignores this basic tenet and contains decryption code which matches exactly that found in a particular known virus. As a result other scanners have produced false positive reports with this particular software.

Responsible vendors realise that their packages must co-exist peacefully with other software (including scanners) in order to reduce confusion. Scanning for viruses is still as much an art as a science and for this reason it makes perfect sense for users to have at least two good scanners in their armoury. In this way, virus infection can be confirmed or dismissed using the different methods which such packages employ.

On the theme of peaceful co-existence, scanners also exist which attach extra bytes to scanned files to indicate that the file has been scanned and found 'clean'. This produces all sorts of problems when another package is used and can even cause problems with the same package as updates become available. Assume for example that a file is infected with a virus which is not known to the scanning package. After (or even during) the scan process, a 'clean' indicator is attached to the file. Now, a subsequent update of the scanner contains details of the hitherto 'unknown' virus. Will it re-scan the affected file? If so, there wasn't a lot of point in adding the 'clean' indicator in the first place - if not, there wasn't a lot of point in adding the update!

Similarly, if part of an anti-virus package is designed to raise the alarm when specific files change in some way, these will be triggered by the added 'indicators' just as if appended virus code had been found. Modification of files in this way is simply arrogant; the best course is to consider the program file as sacrosanct and not to attempt to modify it in any way. As 'virus-aware' software becomes prevalent, appended indicators will cause far more confusion in the future.

Most of these problems are associated with 'passive' scanners that are used to scan disks, directories, files and sectors. However, they also apply in large part to the more recent

development of 'active' scanners which become resident in memory and conduct their searches 'on the fly' as information is passed to and from the storage media (hard disks, floppy disks, tape etc.).

Resident scanners have their own problems; most of them maintain their search patterns and information in memory, and the amount of memory required is growing in direct proportion to the number of viruses identified. Others avoid this problem by continually accessing the disk for specific sections of the identification list as it is required.

The difficulty of dealing with stealth and encryption techniques is causing major headaches for both 'standalone' and memory-resident scanners. The appearance of the Mutation Engine heralds the rapid demise of many scanner programs.

### Smart or Dumb?

A somewhat spurious addition to the debate about scanners has centred around the so-called 'smart' scanners and their (presumably) dumb competitors. The criteria by which a scanner should be judged is its ability to detect and identify virus code accurately, and not whether a particular programmer has used some clever or intricate means of achieving this aim. Obviously, the speed and convenience of a scanning program will remain a major consideration but this should not be achieved at the expense of accuracy. Given two scanners which achieve similar hit rates, only then can due consideration be given to speed, capacity and other factors.

### A Trusted DOS Environment

The generic detection approach must begin to gain ground if we are to continue to provide users with the security they need. Unfortunately, dedicated generic detection has similar problems to specific checking in its absolute need for reliable (i.e. unintercepted) access to the disk.

Even the addition of generic virus checking code to existing programs during the development cycle (thus creating a category of 'virus aware' software) needs some external help to verify the integrity of the operating system. The fact is that when viruses are around, DOS cannot be trusted and extraordinary methods are required to recover the trust that was previously taken for granted.

### The Numbers Barrier and New Directions

Responsible researchers know that scanning programs will run up against the numbers barrier and are aware that a shift in emphasis is inevitable. Over the next year or so, the emphasis will gradually shift from specific-detection to generic and users must be prepared to accept this change. My own opinion is that this is unlikely to cause the complete demise of specific scanning techniques. These will always be required if only to assist the various agencies who need accurately to identify particular virus code - police computer crime units being obvious examples of such agencies.

# TEST PROCEDURES

***Ross M. Greenberg***
***Software Concepts Design***

## Scanner Evaluations - Recent Concerns

Recently, I read a few evaluations of virus scanners with interest. My own scanner, part of the *Virex-PC* anti-virus package, showed up in one evaluation as the most comprehensive and fastest of the scanners reviewed. In another evaluation, my product showed up as eighth, beneath those products previously found to be inferior. In a third, my product wasn't even mentioned. In a fourth, it finished last. This was all for the same version of my code and the other products were also, for the most part, the same releases in each evaluation.

What happened? Are scanner evaluations fundamentally flawed and therefore a worthless means by which to judge the suitability of particular products to the task? Can scanner evaluations be made more effective and more trustworthy?

### How Are Scanners Evaluated, Anyway?

A typical evaluation of a scanner entails the reviewer running the scanners against a disk full of viruses and recording how many 'positive' hits against the virus database occur. Superficially, this *seems* to be the right way of going about things, however, there are many potential pitfalls with this approach.

The first problem is the validity of the virus database. There were some 'viruses' in the virus collection used by the *National Computer Security Association* (henceforth referred to as *NCSA*) in one of its evaluations that contained a file consisting of nothing other than many NOPS followed by a single CD 20 (an exit instruction). This was obviously not a virus: it could not conceivably replicate and was a farcical inclusion in a test suite. Yet scanners were judged against this non-virus. This made me wonder about the scanners which showed 100% effectiveness against that particular 'virus' collection! In July 1991 a *VB* test set included two non-viruses (Nazi and Catman) which were subsequently removed.

Other virus collections have but a single copy of a polymorphic (self-encrypting virus). Some less-than-careful scanner authors might not realize that a given virus was of the polymorphic flavor and might only include a single search string in their detection database; compounding this problem is the fact that the virus might be transferred directly from the virus writer to the evaluator and the evaluator's tests might show a hit for that one example of the polymorphic virus while neglecting to test a scanner against numerous successive generations of the virus. I understand that the current *NCSA* evaluation suite does not include multiple generations of polymorphic viruses; things may have changed by the time you read this, however.

**Two-Faced Evaluations?**

Some evaluators work very closely with a particular vendor. They exchange viruses back and forth with each other, but with no-one else. This virtually guarantees that the vendor who works closely with an evaluator will get better results than those who are not 'in bed' with the evaluator.

Compounding the evaluation problem is that of selective evaluation where a product is tested against known 'losers'. Recently the *NCSA* allowed one vendor to pay for evaluations of his own product and a number of other scanners of his choosing. When the results were published, it was no surprise that the vendor paying the fee came out ahead of the others.

*"This was obviously not a virus: it could not conceivably replicate and was a farcical inclusion in a test suite."*

**The Make-Believe World of The Evaluator**

There are discrepancies between the real world and the evaluator's 'world'. One such discrepancy is inherent in my own code.

For a variety of reasons, my code stores virus 'hits' in an ever-growing malloc-ed array of strings which is later written to the log file. Depending on the memory configuration of the test machine and the number of infected files in the test library, my program will eventually run out of malloc space. Although the screen will show each infected file and the resulting log will show the total number of virus hits, at least one evaluator decided that my program had missed over half of the viruses on their memory-deficient machine with a virus database of many thousand infected files. This was reported in the subsequent evaluation report which showed my program in a poor light. The evaluator did not make a simple phone call to determine the cause of the problem.

In the real world, of course, it is highly unusual to find cases of multiple infection involving more than three different viruses at most, let alone hundreds or thousands - this problem would thus never be encountered by a real user.

**How Long Does It Take?**

Scanner speed, of course, is an important consideration when evaluating scanners. However, testing scanner speed against a heavily infected disk is not a fair test of many scanners. At least one scanner does extensive testing of any target file it suspects is infected. This extensive testing virtually guarantees no false positives and provides a very precise identification of the offending virus, but does slow down the scanning process. In contrast, scanning a clean disk amply demonstrates this scanner's very rapid speed under normal conditions. Scanner evaluations which include speed trials very rarely mention the speed differential when searching through clean and infected files.

Some other considerations should be taken into account. For example, if the scanner conducts a self-test upon start-up this can really skew the results. Also, if memory scanning is performed by one scanner while another scanner doesn't check memory, that 640 K scan can really make a difference. Most vendors include command line options to invoke or disable memory scanning or self-testing: were they used?

Evaluators should ensure that all scanners perform on an even playing field.

**Compressed Files Make A Difference**

*PKLITE* and *LZEXE* are very popular compression add-ons, compressing their host executables and decompressing them at run-time. If a program is internally compressed, it can disguise an infection unless the scanner is capable of decompressing the target file. But 'compressed' viruses are only going to be disguised for the first generation of the virus; subsequent generations will result in 'normal' infections.

Should a scanner scan such compressed files as the normal default option? Or as part of a 'long' scan only? Obviously, a disk with many compressed files will take longer to scan than one with few compressed files, yet the number of compressed files is never printed in scanner evaluations.

**Popularity Versus Completeness?**

Another important consideration: is it important for a scanner to find all known viruses, even when upwards of 80% of the viruses known so far are not found in the wild? One scanner vendor took it upon himself to label some viruses as 'extinct', thereby making his scanner faster by searching for fewer viruses. Although, this might contribute to attractive ad copy, a tighter search algorithm would probably make for better performance and better ad copy too.

It might be better to weight the results of a scanner evaluation based upon the prevalence of those viruses in the wild: missing a Jerusalem infected file is obviously more serious than missing an obscure research-only virus or one destined not to spread too far such as the Whale virus. Of the 1500 viruses known to date, perhaps only 50 have been seen in the wild and are thereby more significant to real world scanner evaluation. I have not seen such a weighting in any evaluation.

**Missing Viruses, But Being Quick About It!**

Many scanners have a 'turbo' mode versus a 'secure' mode. Some scanners will miss a virus or two in turbo mode, but find them in secure mode. What is this ratio, and at what point

can an evaluator declare the turbo mode worthless? Is missing one or two viruses, in particular research viruses, enough to make the turbo mode useless?

The current hype fuelled by prestige viruses such as the Mutation Engine is sort of silly, too. Recently, many of the popular trade press journals have commented on *The End Of Computing As We Know It* because of the utterances of some vendors about MtE viruses. Polymorphic viruses have been with us for over two years, yet suddenly they're new and the columnists, spoon-fed by a few vendors, feel obliged to spread the word about them.

I'm sure that all scanner evaluations will shortly include (and weight disproportionately) MtE derived viruses because of the press outpouring. Yet these viruses are not widespread. The vendor community is forced to react to the evaluation community: the numbers game is the marketing game and the 'my scanner can beat up your scanner' method makes for great ad and editorial copy.

Recently one of my competitors sent out some pithy press releases regarding the Mutation Engine viruses and the supposed ability of his scanner to find all such occurrences. I have a disk full of MtE generations which this scanner misses completely: will the press read that release and report it, or conduct testing themselves to expose its claims as bogus?

*"There is something wrong with evaluations which test old versions of those scanners whose developers refuse to pay extortionate fees in order to get their products regularly tested."*

**Cart before the Horse At A Price?**

The evaluation community is starting to drive the vendor community - and that, in my opinion is a move backwards. The vendor community should only be responding to real-world challenges but is forced, instead, to respond to the *Test-Of-The-Month*.

Speaking of such testing: there are some tests, available for public download, that are driven by money. There is nothing wrong with such commercialised testing, it might even be vaguely accurate.

There is, however, something wrong with evaluations which test old versions of those scanners whose developers refuse to pay extortionate fees in order to get their products regularly tested. There is one evaluation currently circulating in the shareware community which compares a recent release of one of my competitor's products to a six month old release of my own. I release new cuts every six weeks, so how can this be? Simple: I do not pay to have my code evaluated.

Is this a fair evaluation? Compounding the problem is the close relationship the evaluator has with the vendor and the amount of money which changes hands for such evaluations. This questionable form of review process becomes even more suspect when it is realised that one such evaluator receives viruses directly from its largest 'customer'.

**Solutions**

Some solutions to the problems outlined above include:

1. A well thought out plan to create a genuine virus library. Dr Alan Solomon is currently working on a protocol for such a process. Among his suggestions are that the inclusion of a virus in a test-suite should include proof positive that the sample is, in fact, a virus: it must replicate in order to be incorporated. There should be a number of generations of a given virus in the collection: consider the example of a first generation unencrypted but polymorphic virus compressed with *PKLITE* - it would take at least two generational runs for the 'true' virus to materialise for inclusion in the library.

2. Evaluators must be above board: if they have a relationship with a given vendor, that relationship must be declared. *Virus Bulletin*, which commissions evaluations from third parties, has a close relationship with the software vendor *Sophos*: this is acknowledged in reviews and widely known to the anti-virus community. What about other evaluators?

3. If a vendor is allowed to pay for evaluations of its product versus known 'losers', the interim products' 'scores' should be published. If Product A is ranked third and Product B is ranked 15th, the vendor of Product A will publicize the fact that its scanner beats Product B - but the public should also know about the products that beat Product A and the interim products, too. Evaluators, especially those which allow their results to be publicized for a fee, must be responsible for the completeness of their reports.

4. Likewise, evaluators must make sure that only the latest release of each vendor's product is tested: it is worthless for the end-user community to see out-of-date products compared against latest releases.

5. The evaluation process must be better thought out than simply to run a scanner against a collection and see how it fares: what about false-positive testing, secure scanning versus turbo scanning, testing scanner run times accurately, running a test under real-world conditions and so forth?

End-users should drive the anti-virus market. The evaluations of scanners and other anti-virus products should reflect only how well an anti-virus product meets the *real* needs of the end-user community. It is sad to see the evaluator community running the show rather than end-users.

# VIRUS ANALYSIS 1

***Dr Richard Ford***

## Nines Complement - The Accountant's Nightmare

The Nines Complement virus is a memory resident COM file infector which infects programs on execution. The interesting feature of the virus is its interception of INT 17H, the ROM BIOS printer services interrupt, and the effect it has on printed output which is produced using this service. [*Two new variants of this virus are described on page 7. Ed*.]

### XOR 'Encryption'

When an infected program is executed, control is passed to the virus code by means of a JMP instruction placed at the start of the infected code. The majority of the virus code is stored in an encrypted form - the first task the virus undertakes is to decrypt its code by XORing each byte with a number which varies according to the original uninfected program's length. While introducing a degree of variability between successive generations of the virus, this encryption poses only a minor obstacle to disassembly - once decoded, this virus is straightforward to analyse and the encryption routine is trivial: no competent analyst will be thrown by such a simple trick.

The very minor advantage gained from this 'encryption' routine is that the search pattern for file recognition is of a limited length.

The virus checks the contents of the CMOS at locations 36, 37 and 38. The rationale behind this is not clear because there are such large variations in CMOS contents between different machines. The virus searches for the ASCII string 'Tys' in the CMOS, and, if this is found, it returns control to host file.

Next, the virus checks to see whether it is already resident. To do this it executes INT 21H function 4B99H - if the virus is resident the value DAADH is returned in AX, and once again, the host program is executed.

The INT 21H and INT 17H interrupt vectors are obtained (oddly, by using DOS interrupt INT 21H function 35H) and the virus copies itself, plus the host program to a higher memory location. The virus is then copied to the original code segment and the INT 21H and INT 17H interrupt vectors are set to point to addresses within the virus. This is achieved by using the DOS interrupt INT 21H function 25H (Set Interrupt Vector) - this is very much a text book technique and a very 'nice' way to program. How touching it is to see a virus writer take such care not to break any of the rules of programming under DOS. At this point, a new Program Segment Prefix is created at the start of the current code segment and control is returned to the copy of the host program.

### The INT 21H Routine

The only INT 21H function intercepted is INT 21H function 4BH (Load and Execute) - all the other INT 21H calls are passed straight to the DOS INT 21H handler. The infection routine is very simple. The target file's attributes are stored and then reset to r/w access, not hidden etc. The file is opened and checked to see whether it is already infected. If the file starts with either an M or a Z it is assumed to be an EXE program and the infection routine is terminated and control passed to the DOS INT 21H handler. The date and time of the creation of the target file are read in to memory, and a check is made to ensure the file does not exceed FFFFH in length.

If all these conditions are met the virus allocates a memory block via INT 21H function 48H and encrypts a copy of itself in this block, before adding itself to the end of the target file. The routine then resets all the target files' attributes and passes control to the DOS INT 21H handler. If an error is detected at any point during this process the virus aborts its infection routine. The only point of any interest at all in the infection routine is the way some of the DOS INT 21H calls are set apart from code and CALLed - why anyone would write code like this for such a small program is beyond me.

### The INT 17H Routine

The INT 17H routine appears to be this virus' raison d'être. Calls to the printer via INT 17H are intercepted, and if the INT 17H call is to send a character to the printer, the ASCII value of that character is read. If the character is a number the ASCII value is altered so that the number 9 becomes 0, 1 becomes 8, 2 becomes 7 etc., and the call is passed on to the printer. If the character to be printed is not a number, it is passed on to the printer unaltered.

### Nines Complement

| | |
|---|---|
| Virus Name | Nines Complement |
| Type | Appending Parasitic on COM files. |
| Infective Length | 705 bytes |
| Intercepts | INT 17HH ROM BIOS printer services |
| | INT 21H for infection route via 4BH |
| Trigger | Swaps digits 0 to 9 around when printing using INT 17H |

System recognition via 'RU There?' call which returns DAADH in AX when INT 21H 4B99H is called

File recognition

```
E800 005B BE11 0003 F3B9 AA02 89F7 AC30 D8AA E2FA
```

# VIRUS ANALYSIS 2

## November 17th Strikes In Italy

A recent report from Italy concerned a new virus called November 17th and although several machines were affected at one site, no other reports have been received to date.

This virus is quite primitive, a resident infector of COM and EXE files through the usual process of appending its code to the end of the target program file. It has a trigger routine that will trash the default disk under certain conditions between 17th and 30th November (inclusive). The code appears to be original, i.e. not copied from another virus, and shows signs of being written by a competent programmer.

### Operation

When the virus is first executed, the word at 0000:020CH (the INT 83H vector address) is checked for a value of 5856H. This is not an address, simply an indicator value to show that the virus is resident, it may even represent the letters 'XV'. If the virus is not resident, a special flag is cleared within the virus code, and the whole of the code is relocated to an offset of 100H as high as possible within the currently allocated memory block. Once there, the code hooks its own handler routines into the INT 21H and INT 09H interrupts and finally returns control to the host program.

The INT 09H handler (invoked during each keypress) simply maintains a counter, decrementing a number as each key is pressed and testing it for zero. The counter is initialised to 500 when the virus is first installed and a special flag is set when zero is reached.

The INT 21H handler intercepts only function requests 3DH (Open file), 43H (Change Attributes) and 4B00H (Load and Execute). A special effort is made to avoid intercepting requests to open a file for Read Only access. Similarly the handling routine is made re-entrant by the provision of a status flag to indicate that certain areas are busy.

Once inside the handling routine, the experience of the programmer is readily observable in the way that checks are conducted for COM or EXE extensions as well as some additional checks that avoid invoking the infection routines in the presence of programs with the names of SCAN.EXE and CLEAN.EXE (i.e. *McAfee Associates*' software).

### Infection

Only COM files of less than 60,001 bytes in length are infected but while there is no limit on the size of EXE files, special care is taken to avoid infecting EXE files where the image size and the true file size differ (as with many *Windows* files). The target file has the first part of it read into a buffer (24 bytes for EXE files, or 8 bytes for COM files) and this is then checked to verify that the target is indeed an EXE file (with the 'MZ' header) and then to see whether the file is already infected. This is done by examining a particular word in the header for a value of 5A4DH (exactly like the 'MZ' header). The word examined is the Checksum field in the header for EXE files, or the 5th and 6th bytes of a COM file. If the target is suitable, the original header information is stored within the virus, the target file has its header modified and 855 bytes of virus code are appended to the the file.

### Trigger

The trigger routine depends upon a key counter and the system date. If the key counter has reached zero (i.e. after 500 or more keypresses since the virus went resident) and the system date is between 17th and 30th of November (inclusive), then a simple trigger routine is invoked which identifies the current logical drive and writes garbage to the first 8 sectors using the INT 26H absolute sector write function call.

### November 17th

| | |
|---|---|
| Virus name | November 17th |
| Aliases | 855 |
| Type | Resident, Appending Parasitic on COM and EXE files (including COMMAND.COM) |
| Infection | COM files less than 60,001 bytes, and all EXE files. |
| Recognition | |
| File | 'MZ' in Checksum field of EXE file headers.<br>'MZ' as 5th and 6th bytes of a COM file. |
| Hex Pattern | `3D75 04A8 0174 1180 FC43 740C 3D00 4B74 0FE9 2602 59E9 F001` |
| System | 5856H value in 0000:020CH indicates virus is resident. |
| Intercepts | INT 21H for infection and detection of trigger conditions.<br>INT 24H for internal error handling<br>INT 09H to maintain counter which sets trigger flag |
| Trigger | Overwrites first 8 sectors of default logical drive with garbage, uses INT 26H Absolute Disk Write |
| Removal | Specific and generic disinfection is possible. Under clean system conditions, identify and replace infected files. |

# PRODUCT UPDATE

## Fastback Plus v.3

No matter how good other anti-virus precautions are, there is always a possibility that they may fail, and allow a virus to trigger. Backups provide a useful safety-net.

One of the best known backup programs for the PC is *Fastback Plus*, which *VB* last looked at two years ago. *Fastback Plus* creates backups to almost any MS-DOS device, and provides data compression, encryption, error correction, and facilities for specifying exactly which files should be backed up. When *Fastback Plus* was last reviewed, it was available as v.2.10. It is currently available as v.3.02 with many improvements and the following new features: a *Windows*-like interface with multiple overlapping windows, mouse support, backup manipulation according to a timed schedule, DES file encryption, and *Novell* network support.

Installation is easy. The user is given the choice to use *Fastback Plus* with MS-DOS, or whether a *Windows* installation should also be performed. Installation worked correctly in both cases on the hardware used for this review. Note that *Fastback Plus* v.3.02 executes as a DOS program under *Windows*. It is '*Windows*-aware', but is not itself a *Windows* program. Unsurprisingly, the user interface operates more slowly under *Windows*.

The user interface of *Fastback Plus* has been changed to provide pull-down windows, mouse operation, and a dialogue-box style of operation. In common with all programs designed for *Windows*, use of a mouse is not mandatory, but the whole user interface is geared towards this style of operation. To help users through the myriad options, 'Express' menus are available which offer salient features in a simple easy to understand format. This works well, and allows naive users to manipulate backups with minimal knowledge.

*Fastback Plus* contains a built-in macro system, which is used for automatic backups. A scheduler operates according to a pre-defined timetable, and macros are included which create, test and restore 'standard' types of backup. The macro system permits commands to be replayed at the same speed as they were entered: a useful feature for teaching new users.

Although *Fastback Plus* normally uses standard MS-DOS floppy disks, if the user so desires, it can provide up to 11% extra capacity per floppy disk. For example a 3.5 inch 720K floppy disk can be made to hold 800K.

I tested the speed of creating backups by measuring how quickly *Fastback Plus* could fill a 1.44 Mbyte floppy disk while taking a complete backup of my hard disk. Version 2 of *Fastback Plus* filled each floppy disk in 40.2 seconds. Version 3 of *Fastback Plus* took 45.5 seconds to carry out the same test when executed under DOS, and 41.9 seconds when executed under *Windows*. This variation is not significant, the only fair conclusion being that not much has changed, and the backup creation process on my PC is probably limited by the speed at which the floppy drive operates. These figures convert to a backup speed of approximately 2.16 Mbytes per minute. When data compression is taken into account, this figure is more like 3 Mbytes per minute; an impressive rate of backup creation.

The compression technique has improved. Version 2 of *Fastback Plus* requires 21 disks to backup my hard disk, while version 3 requires just 15 disks for this task - a saving of about 25% (the manual modestly claims that data compression has been improved by up to 15%). This improvement in the results is exaggerated by the presence of many Megabytes of text files on the hard disk which are relatively easy to compress. Even so, I'm impressed at these savings in disk space. A trade-off can be made in data compression between disk space used, and time taken, but the floppy disk write time masked this variation during testing. This would not be the case if backups were made to a device capable of writing at a higher speed, such as a Bernoulli disk, or an optical drive.

The manual states clearly that v.3.0 of *Fastback Plus* cannot restore backups made with v.2.09 or earlier due to a new compression technique used from v.2.10 onwards. Users with large stores of old backups should be aware of this problem.

I liked v.1 of *Fastback Plus*, v.2 was a substantial improvement, and the features introduced with v.3 add significantly to the available security features. It is one of the best backup programs around, but changing the program to make it *Windows*-aware, has slowed down the operation of the user interface. These changes have not had any detrimental effect on the actual backup facilities, which remain excellent. My reservation centres upon doubts about whether the lemming-like dash towards *Windows* software is actually *progress*. KJ.

**Technical Details**

**Product**: *Fastback Plus*

**Developer**: *Fifth Generation Systems Inc.*, 10049 N.Reiger Road, Baton Rouge, LA 70809, US, Tel: +1 (504) 291 7221, BBS: +1 (504) 295 3344

**Vendor**: Sold by many (most?) software dealers.

**Availability**: IBM PC/XT/AT, PS/2, or compatible with 640K of RAM (512K available RAM), MS-DOS version 3.0 or higher, and a hard disk.

**Version Evaluated**: 3.02

**Serial Number**: 133-0981348, supplied on 1x3.5 inch and 2x5.25 inch floppy disks.

**Price**: £120+VAT

**Hardware Used**: Toshiba 3100SX, battery powered laptop portable with a 16MHz 80386 processor, one 3.5 inch (720K) drive, and a 40 Mbyte hard disk running under MS-DOS v5.00.

# PRODUCT REVIEW

***Dr Keith Jackson***

## ASP Integrity Toolkit

The *Integrity Toolkit* was developed by Fred Cohen in the USA. It is the most complicated anti-virus product that I have reviewed. 'Complication', in this context, denotes the breadth of available facilities, rather than difficulty of use. That said, I did encounter teething problems, and suffered one heart-stopping moment, while evaluating this product.

### Available Features

The list of *Integrity Toolkit's* features is so long that is very difficult to summarise them.

The product rightly considers anti-virus features to be but one aspect of Information Security. When the *Integrity Toolkit* is installed with access control enabled, a user must log in using his own ID and password. While a PC is in use, audit trails are maintained and users are constrained to a preset suite of facilities.

Data stored within the PC can be encrypted. This can be done either by encrypting only the File Allocation Table (FAT), or by encrypting all data stored on the hard disk. The latter option is much slower but more secure. In addition, a program is provided which modifies the hard disk so that it cannot be accessed when the PC is booted from a floppy disk (this process can be reversed). Integrity is preserved by using a checksumming program which takes 'snapshots' (fingerprints) of all important files and areas of the hard disk. These checksums can be tested both at boot time, and dynamically before execution of a program.

The *Integrity Toolkit* can also create online backups which can be used for automatic and transparent recovery, so that corrupt programs can be replaced without hindering the operation of the PC. Online backups can be encrypted or stored on another DOS device (perhaps remotely across a LAN). Custom installation with your own text messages is possible.

Specific features which protect against viruses are a scanner, an 'Execute only' mode of operation which prevents modification of any executable file (as such files cannot be read from, or written to, this also acts as a simple copy-protection measure), and format prevention to prevent a disk from being accidentally trashed. A 'Trace Trap' feature prevents file execution from being examined with a debugging tool. Files can be securely deleted by being overwritten. Trusted facilities can be turned off dynamically so that tasks such as changing passwords and taking backups can be performed.

### Customisation

As if the features described in the previous section were not already enough, their application can be tailored in almost any desired manner. This is done by using a menu driven program which can manage, use, explain, analyse, tailor and remove the facilities available within the *Integrity Toolkit*. An impressive list by any standards.

The type of access control used by the *Integrity Toolkit* can be either Two Type, POset, or MilSpec. The manual contains a cursory explanation of what is offered by each method, but goes nowhere near the level needed to explain such terms to a uninitiated user.

When installing an integrity shell, there are many tradeoffs that can be made. Four different methods of checksum calculation are offered ranging from cryptographic algorithms, down to mere inspection of a file's size and its time of last alteration. The type of file in which the checksums are stored can also be selected - sequential storage, or storage in a hashed form. Hashing is faster, but uses more disk space (see Figure 1.).

| Algorithm | Checksum calculation (seconds) | Checksum file size | Execution speed |
|---|---|---|---|
| Big / Hashed / Slow | 204 | 320K | 3K/sec |
| Big / Hashed / Fast | 202 | 320K | 3K/sec |
| Big / Hashed / Trivial | 64 | 160 bytes | 9K/sec |
| Small / Hashed / Slow | 134 | 160K | 4K/sec |
| Small / Hashed /Fast | 131 | 160K | 4K/sec |
| Small / Hashed / Trivial | 96 | 160K | 6K/sec |
| Sequential / Slow | 62 | 176 bytes | 10K/sec |
| Sequential / Fast | 61 | 179 bytes | 10K/sec |
| Sequential / Trivial | 28 | 179 bytes | 21K/sec |

Figure 1. Four checksumming algorithms are offered in the *Integrity Toolkit*. Calculation times to checksum 593 Kbyte in 14 files are shown. The checksum database files vary in size from 320 Kbyte to just 160 bytes. The 'slow' algorithm was published in *Computers & Security*, the 'fast' algorithm is proprietary while the 'trivial' algorithm is a manipulation of file size, time stamp etc. This table graphically illustrates the configurability of the *Integrity Toolkit* - there is even the option for the user to introduce his own algorithm. Note that these algorithms do not comply with ANSI or ISO standards.

Any installation beyond the factory-set default will require a lot of thought. I think that more explanation is needed to install the *Integrity Toolkit* than that provided.

### Documentation

The manual is an 89 page A5 booklet, packed full of information, but rather unyielding to new users who are unfamiliar with the facilities offered by the *Integrity Toolkit*. My main criticism is that it is terse and would be improved by much more explanation couched in simple terms. As an example, consider the term 'POset' which appears on page 31. Look this phrase up in the glossary and the definition given is 'Partially Ordered set'. Most users of the *Integrity Toolkit* won't know what this means and more explanation is called for. I was also taken aback by the lack of explanation of most of the error messages, and the lack of a specific section providing advice on what to do if things go wrong.

A glaring omission in the documentation was the lack of any telephone number to contact for further support. I encountered problems during installation and was in the fortunate position of obtaining Fred Cohen's number from *Virus Bulletin*. What does a mere user do? The *Integrity Toolkit* manual had information from a Danish distributor attached (see *Technical Details*), but this too omitted to include a phone number.

A section entitled 'Making Protection Decisions' discusses the tradeoffs between the various facilities on offer. *Decisions, Tradeoffs and Questions* discusses each of the following: locking the hard disk against floppy boots, snapshot methods, access control, integrity shells, online backups, pattern matching, audit trails, etc.

My gripe about the documentation is not about its technical content, which is excellent, but about its failure to help mere mortals understand what is a very complicated product indeed.

### Installation

The *Integrity Toolkit* requires that it is installed on a hard disk and as several Mbytes of files are either copied across from floppy disk, or created during the installation, this requirement is unsurprising. The *Integrity Toolkit* was provided on a single 3.5 inch, 1.44 Mbyte, floppy disk, containing just over 1 Mbyte of files. Given that older PCs (e.g. my normal test machine) can only ever use 720 Kbyte 3.5 inch disks, the high density disk prevents installation on such machines. *VB* did not request a 1.44 Mbyte disk, it just came that way, and I would suggest that 720 Kbyte disks (however many are required) would cause fewer installation headaches. [*720 Kbyte diskettes are available upon request. Ed.*]

Installation seemed deceptively simple. Insert the floppy disk in drive A:, execute a program called MIT, and after a few seconds of activity, a message saying 'Integrity Toolkit not installed - install it? (Y/N)' appeared on the screen, coupled with a series of insistent bleeps. Being a simple soul, I answered 'Yes' to this question. This proved to be a mistake.

The *Integrity Toolkit* asked for a superuser password, advised me to use a hard reboot if its own soft reboot failed, and said that it was creating the required snapshots of my hard disk. As an aside, the manual states that it will prevent a user from having an unacceptably weak password. I can vouch for this, it took me six attempts to dream up a password that the *Integrity Toolkit* would accept. Nowhere in the documentation does it explain its internal rules for strong passwords. The user just has to guess what may or may not be acceptable.

After installation, the *Integrity Toolkit* rebooted my PC. A few seconds later, a strongly worded warning message appeared stating that this was a system for authorised users only, coupled with a short DOS error message stating 'Error loading command processor, SYSTEM HALTED'.

No matter whether I booted from the hard disk, or from a floppy disk, my hard disk did not exist. The DOS message 'Drive C: does not look like a valid DOS disk' was always displayed when I tried to access it. Even though I lay claim to be the world's greatest bore about taking backups and follow my own advice religiously, such an event is guaranteed to get the adrenalin flowing. No matter what I tried, I didn't have a hard disk anymore as far as DOS was concerned. It seemed likely that the *Integrity Toolkit* had altered the Partition Table and thereby prevented DOS from recognising the disk.

At that point I hoisted the white flag of surrender and telephoned Fred Cohen. After querying whether I had read the first page of the installation section of the manual (my answer was a sheepish 'Probably not'), I was reminded to read the appropriate sections of the manual if I used either a memory manager, networking software, a software disk cache, or other protection software. I use both a memory manager, and a disk


Teach Mode refusing to take a leak! Teach mode is an interactive help system which explains the features of the *Integrity Toolkit* and the threats to data integrity.

cache, so I failed on two counts. Even after reading the appropriate sections I'm none the wiser, as pared down to the bones it states four specific instances where problems are known to occur (only one of which could possibly have been relevant to my PC), and states 'If default installation fails, there is a chance you will have to use the recovery techniques listed earlier to regain access to your system'. This may well be true, but is it good enough?

After I described the problem further, Fred Cohen's near instantaneous response was to say 'Have you read page 29'. This page explained how to use the *Integrity Toolkit* to put the Partition Table back on my hard disk.

While using this option, I realised that had I answered 'No' to the original installation query, I would have seen the excellent menu options which would have provided enough help to avoid the above trouble.

The response to my telephone query was excellent and it got me out of trouble. Ten out of ten for support. However, to harp back to a previous point, just what should I have done without access to Fred Cohen's personal telephone number? [*The developer states that the Integrity Toolkit is intended for corporate use only and that support telephone numbers are arranged as and when the product is installed under supervision. Ed.*]

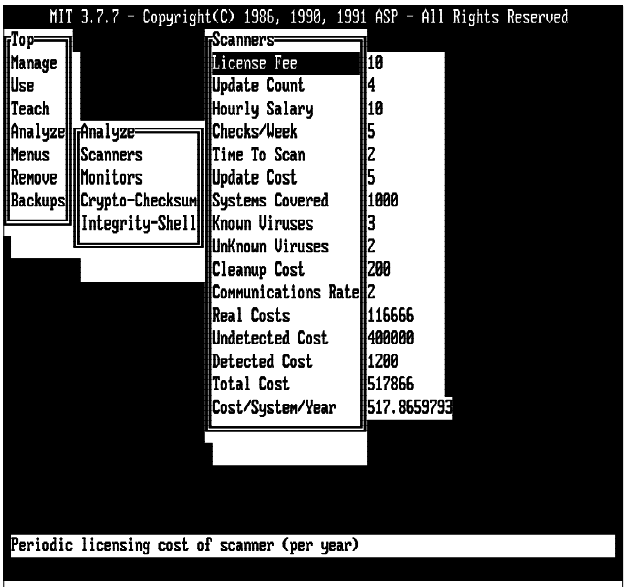
The error message seen was not explained anywhere in the documentation. My guess that the *Integrity Toolkit* could not find COMMAND.COM because it was not in the root directory proved correct. Fred Cohen assures me that the *Integrity Toolkit* can be persuaded to look elsewhere for the command processor, but I prefer my simplistic cure of placing an extra copy of the command processor in the root directory. To escape my problems, I had to put together the disparate thoughts that manually turning off 'BOOTLOCK' protection might be the cure for my 'Cannot find command processor' message - a deduction beyond the grasp of many PC users.

My main gripe about all this is that it matters not what options the *Integrity Toolkit* offers amongst its myriad menus if the first question during installation states that the product is not currently installed, and would I like it to be installed? The answer to this question will usually be yes. The default option during installation should be to go into the *Integrity Toolkit's* splendid 'analyze' mode (see below), which could then impart information about setting up the program correctly.

### Legal Shenanigans

After the problems described above, I was unhappy to find that the manual contained the following paragraph:

> *We make no warranty either expressed or implied as to the suitability of this product for any particular purpose. We disclaim all responsibility for direct, consequential or other damage resulting from its use.*




Analyze, a feature unique in anti-virus software, is a cost-benefit analysis program to determine the most cost-effective approach to combating viruses.

It is normally not possible to disclaim all responsibilities in law. Such legal shenanigans should have no place in a serious computer security product. It is right and proper to limit responsibility; it is not acceptable to just disown it.

### A Difficult Product To Review

After a straightforward description of the features offered by the *Integrity Toolkit*, I'm faced by the awesome thought 'How on earth does anyone review this in 1500 words?'. In reality such a task cannot be done at a level of detail which would do the product justice, this article is thus a mere 'taster' rather than a complete review. I seriously doubt that a product as complex as the *Integrity Toolkit* can be 'reviewed' in the normal sense. There are options to tailor everything, literally everything, which from the security viewpoint is excellent, but it does mean that any criticism could always be deflected by claiming that it only applied to one particular installation.

In summary, the parts of the *Integrity Toolkit* that I tested performed as expected. Beyond minor niggles there is not a lot to report. However, I would like to go into detail about just a few things: the teach mode, analysing the effect of various security options, and scanning. The latter simply because it provides one of the few facilities in the *Integrity Toolkit* which can actually be measured.

### Teach Mode

Teach mode is a structured interactive help system which allows a user to browse around and learn about the various security features. It operates using a suite of text files, which a

security administrator could tailor to refer to specific 'local' features. This is an excellent idea, and one that other like-minded programs would do well to follow.

If I had discovered Teach mode before I encountered installation problems, I would have understand the nature of the problem a little better, instead of just making guesses as to the cause of the problem (however correct they turned out to be!)

### Analysing

I have never before come across a program which analyses the financial effects of using various anti-virus methods! It is very thought provoking to have the consequences of using virus scanners, virus monitors, cryptographic checksumming, and an integrity shell explained through the use of examples that have been worked out in detail, to illustrate their effectiveness, their cost, and their impact on PC operation.

The input data to these examples can be altered as desired, and the user can play 'what if', to see the consequence of his actions. It is possible that some of Fred Cohen's original assumptions about the costs of various anti-virus features are unduly pessimistic in some directions and perhaps unfair to some techniques. However, the point is that the analyse mode of operation does allow information to be collated with differing input data. Discussion can then follow on an informed basis rather than by guesswork.

I like this 'analyze' feature very much and would recommend that other products incorporate a similar feature. Perhaps then, manufacturer's complaints about *VB* reviews would become more objective and contain less 'marketing-speak'.

In fairness, this cost-benefit analysis feature deserves a complete article in itself. The precepts upon which Cohen bases his calculations will be found in his book *A Short Course on Computer Viruses* (see *VB*, June 1991, p. 23)

### Scanning

The *Integrity Toolkit* manual is scathing about the use of 'scanner' type programs to detect viruses. In states '*WARNING: Scanners are not sound for virus defense, and are not generally considered a long-term or effective defense*'.

Ironically, although the *Integrity Toolkit* is so disparaging about scanning for viruses, it still provides scanning features, which perform strikingly well. The scanner (Fridrik Skulason's *F-PROT* ) incorporates disinfection/deletion of infected files, searching within compressed files, a heuristic (guessing) analysis mode, and a wealth of detail about virus variants.

Of the 183 viruses which comprise the *VB* standalone test-set, *F-PROT* detected them all. This is a rate of detection which most commercial products fail to achieve!

As for speed of scanning, *F-PROT* is not the fastest around, scanning a hard disk containing 2.3 Mbytes (84 files) in 24 seconds, but it is quite fast enough for normal use. For the record, *Dr Solomon's Anti-Virus Toolkit* performed the same task in 9 seconds and *SWEEP* from *Sophos* in 16 seconds. [For an evaluation of *F-PROT* see *VB*, December 1991, pp. 21-23.]

### Conclusions

This review has barely begun to address the numerous features offered by this product. My conclusions are therefore based on some initial observations.

Many vendors go to great lengths to test software on many different computer systems - *Microsoft* was rumoured to have 3,000 beta-testers for v3.1 of *Windows*. I accept that such extensive beta-testing is beyond Fred Cohen's capabilities. However, without such extensive testing, telephone support, and far more comprehensive documentation, I would not recommend the *Integrity Toolkit* in its current form for the general PC user.

Conversely, if you are a computer security professional, run, don't walk, to obtain a copy as the *Integrity Toolkit* provides a stunning breadth of facilities and puts to shame most other products which merely scratch the surface of the problem. The Integrity Toolkit is, to say the least, *extremely* thought provoking!

*ASP* acknowledges that the *Integrity Toolkit* is a decidedly corporate product made available only to organisations which manage their environments. In conjunction with the installation of the software, the developer performs education, training and customisation to provide a near-bespoke solution to the individual customer's particular needs. In most cases, a LAN manager conducts installation of the product assisted by an *ASP* representative.

I will continue experimenting with the *Integrity Toolkit*, after this *VB* review. I don't often say that about a product!

#### Technical Details

**Developer**: Fred Cohen, *ASP Press*, PO Box 81270, Pittsburgh, PA 15217, USA.

**Vendor**: *Sikkerheds Radgiverne*, Gassehaven 52, Holte, DK-2840, Denmark.

**Availability**: Not stated

**Version Evaluated**: 3.7.7

**Serial Number**: None visible

**Price**: Corporate licences only. Evaluation copies are available for $89 and include the book *A Short Course on Computer Viruses.*

**Hardware Used**: A Toshiba 3100SX laptop with a 40 Mbyte hard disk, 5 Mbytes of RAM, a 16 MHz 80386 processor, and a single 3.5 inch floppy disk drive.

# END-NOTES & NEWS

*Intel Corporation* has announced that ***Novell Inc.* has purchased a corporate-wide licence for *Intel's LANProtect* software**, a file server based anti-virus utility. *LANProtect* runs as a NLM (*NetWare* Loadable Module) on *Novell* servers. *LANProtect* has a recommended list price of £606.00 per file server. *Intel* will provide registered users with unlimited free updates to its virus pattern library. Information from *Intel UK*. Tel 0793 696000.

*Opal UK* has released a **virus-proofed computer** and is planning to release **virus detection software called *Protector*** in the next few months. The company came in for some criticism recently for writing viruses to test the efficacy of its products. Tel 091 491 1234.

*S&S International* has announced a **data recovery breakthrough** in the form of the 'Data Signal Analyser'. The company claims that the new technique makes mainframe data recovery possible and that it can even restore overwritten data by subtracting the signal of the latest data and analysing the signal that remains which is residual from the previous data. Tel 0442 877877.

In an unexpected move, *Ontrack*, the US data recovery service and North American distributor of *Dr Solomon's Anti-Virus Toolkit*, has established a **sales office in the United Kingdom**. *Ontrack Data Recovery Europe Ltd* is based in Kingston-upon-Thames. Tel 081 549 3444.

***Approaching Zero***, to be published by *Faber & Faber* on July 6th, is a **new book which examines the computer underground**. Authors Bryan Clough and Paul Mongo promise previously unpublished material plus in-depth interviews with some of the shadier characters in the world of virology, hacking, phreaking etc. RRP £14.95. ISBN 0-571-16546-X.

***Sophos Ltd* has resigned from the *NCSA Anti-Virus Product Developers* (*AVPD*) scheme**. Dr Peter Lammer said that *Sophos* could not devote to the *NCSA* the level of interest it sought, and requested that *Sophos* be removed from 'NCSA, ICSA, APVDEFGH etc.' In a separate statement *Sophos* has announced the appointment of *ACT Inc* as its US distributor, bringing the number of its overseas distributors to nineteen.

The formation of the ***Virus Security Institute*** was announced in May. This non-profit US organisation aims to provide factual and meaningful information, to counter misinformation and to develop effective product testing protocols. Tel 607 326 4422.

London based ***Menorah Software*** has been **appointed UK distributor** for the Israeli product ***Anti-Virus Plus*** from *IRIS Software*. Tel 081 883 4269.

***Virus Bulletin's* Second International Conference**, Edinburgh, Scotland, 2nd-3rd September 1992. Tel 0235 531889

*IBM UK Ltd* is holding a **Virus Management Course** on June 24th and a **Virus Hands-On Course** on June 25th in Bristol. Tel 081 864 5373.


## 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