ISSN 0956-9979 FEBRUARY 1994

VIRUS BULLETIN

THE INTERNATIONAL PUBLICATION ON COMPUTER VIRUS PREVENTION, RECOGNITION AND REMOVAL

Editor: **Richard Ford**

Technical Editor: **Fridrik Skulason**

Consulting Editor: **Edward Wilding**, Network Security Management, UK

IN THIS ISSUE:

• ***OS/2* spreading.** The latest edition of the hacking magazine *40Hex* contains source code for an *OS/2*-specific virus. How much of a threat does the operating system face?

• **Acorn viruses: a growing problem.** The *Acorn Archimedes* is an increasingly popular choice of computer for schools and colleges. If your organisation owns any *Acorn* computers, turn to page 15.

• **Computer law in action.** New laws are being implemented, which may (or may not) make a difference to authors of both viruses and scanner programs. Do virus authors hold copyright on a hex pattern extracted from their handiwork? See page 13.

CONTENTS

EDITORIAL

Evolving Ideas

When a user calls *Virus Bulletin* and asks for advice on anti-virus software, what he almost certainly wants is advice on virus *scanners*. Most reviews and discussions of anti-virus techniques centre around the scanner, and it is by far the most common safeguard used against viral attack.

Researchers in the industry would doubtless point out that a scanner, by its very nature, can only defend against known viruses: it operates by checking for a predetermined pattern or sequence. Therefore, for a high level of protection, users should employ generic virus detection techniques, such as an integrity checker. Preliminary feedback from the *Virus Bulletin* Readers' Survey, however, indicates that for many companies, the *only* line of defence is a virus scanner.

“ The ability of the average company to defend itself from an attack by an unknown virus is nil ”

As the number of bizarre 'one-off' viruses continues to grow, it is becoming increasingly important that good security practices are followed. The development of *Windows*- and *OS/2*-specific viruses means that it is vital to amend working practices to fit in with the threat: it should no longer be considered sufficient to check the integrity of a machine from a *Windows*-based package; a clean boot is necessary. The rise in virus complexity, coupled with the increased occurrence of hitherto unknown viruses on unsuspecting users' machines, has led to a situation where the well-prepared IT Manager must be ready for a virus which will not be detected by the scanner of his choice.

There is, however, a momentum which needs to be overcome before policy and procedure can be amended. Firstly, new dangers need to be recognised. This is difficult in the case of a virus threat: the chaos which can be caused by a virus attack is difficult to believe unless experienced at first hand. Secondly, the risk to a particular company is perceived to be very small - but in the event of a major outbreak, the results can be devastating.

If (and it is not beyond the realms of possibility) one of the virus writing groups were to release 100 new polymorphic viruses into the wild at one time, without making any announcement to the anti-virus industry, the result would be universal panic… because many companies are totally reliant on scanner technology. The ability of the average company to defend itself from attack by an unknown virus is nil.

An interesting and often eye-opening test of an anti-virus policy is to consider what would happen if a disk which contained a hitherto unknown virus were sent to a large company. Clearly, the disk would pass through any static disk checks, after which the virus would spread unchecked. If it were well written, it is conceivable that the virus could remain undetected for months, until it triggered. Hardly an acceptable proposition for an IT-dependent culture.

At the present time, the computer industry has, to a very high degree, placed all of its eggs in one basket. This is poor practice from a security point of view: companies now rely on computers and their integrity to do business. The computer systems of a large company are, to a very real extent, the foundations upon which the rest of the business is built up.

This is not to say that virus scanners are dead. However, everyone should bear in mind that they can only detect known viruses. Users will not awake next week in a world where the scanner is useless, nor next month, nor (probably) next year. However, what *will* happen is that scanners will gradually become less reliable. Whether vendors are capable of keeping up with the large numbers of viruses received each month is immaterial: there will always be virus authors who deliberately release their creations into the wild, without sending a sample to a researcher.

There is tremendous benefit in techniques which provide a reduction in the threat from both known and unknown viruses, such as checksumming and disk authorisation software. As the balance in the anti-virus world shifts, users must weigh up the strengths and weaknesses of each line of defence, and be prepared to change their anti-virus policy accordingly. *Tempora mutantur, et nos mutamur in illis*. Those who ignore this advice do so at their peril: remember what happened to the dinosaur.

NEWS

OS/2 Virus Developed

The first *OS/2*-specific virus has been developed by members of the US-based *Phalcon/Skism* group. The virus, named (rather unimaginatively) OS/2Vir_1 is a primitive overwriting virus, which takes advantage of the operating system's API functions to search for, open, and infect files.

OS/2Vir_1 poses a minimal threat to the *OS/2* community in its current form; the danger lies in the fact that the virus was published with its full source code in the underground virus-writing magazine *40Hex*. This code, which 'should definitely be helpful for anyone who wants to write viruses in *OS/2*', is claimed to have been written by an individual named Arthur Ellis. Full assembly and link instructions are included with the virus.

Steve White, from the *IBM TJ Watson Research Center*, explained that the discovery of an *OS/2*-based virus has been expected for some time. 'It is not surprising that we are now starting to see *OS/2*-based viruses - we have already seen that any platform which has a large number of users is likely to be targeted by the virus writers.'

Although the virus is a nuisance for the anti-virus community, it will not, according to White, pose much of a threat to the end user. 'The virus is not going to spread in the form in which it was published, and it is not a very helpful example of how to write an effective *OS/2* virus, as it is neither memory-resident nor stealth. Unfortunately, because the source code is freely available, the virus will have different binary forms, depending on which compiler was used. This means that it could be difficult for any one manufacturer to detect every possible variant of the virus.' ∎

Cover Disk Confusion

A report was recently received about a virus alleged to have been found on a magazine cover disk: on 31 December 1993, an article appeared in the Norwegian newspaper *Verden Gang*, claiming that a user had found a virus on the disk belonging to the first issue of the computer magazine *PC-Gamer*, produced by *Future Publishing.*

Virus Bulletin contacted the journal and spoke to its editor, Gary Whitta, who said that they had received no reports concerning the rumoured virus. They had published, he declared, over 90,000 copies of the magazine worldwide, and had previously not even been aware of the existence of the Norwegian article. He assured *VB* that every Master Disk goes through a series of rigorous checks, both by the Editor and by the disk-copying laboratories.

VB contacted *Verden Gang* and spoke to the journalist who wrote the article, Jorunn Stølan. She said, 'I contacted *Narvsen*, the Norwegian distributors of *PC-Gamer*, who said they had no information about the incident, nor had they had any other complaints. In the past, they have had other instances, on other magazines, where one isolated disk has been found to be infected. This could usually be traced back to a user's infected computer, and not to the cover disk. In my article, I was at pains to make it clear that it was merely a possibility that the virus had come from *PC-Gamer*.'

Steve Carey, of *Future Publishing*, commented: 'We refute this allegation totally. Given that this is indeed the only report, I am confident that it is spurious, and has no basis in fact. It is most probably down to misunderstanding on the part of the consumer, and misunderstanding and ignorance on the part of the newspaper concerned.'

This situation is far from rare: user buys magazine, inserts non-write-protected cover disk into (infected) computer, and executes a program, thereby infecting the disk. His first reaction is to assume the disk was infected prior to purchase.

There is a simple solution to this dilemma: the cover disks could be shipped permanently write-protected. Although this would be no guarantee that an infected cover disk had been sent out by the manufacturer, it would at least prevent the all-too-common chain of events described above.

The incident highlights the problems associated with gathering evidence after a virus attack, in an attempt to trace the source of infection. When carrying out such an investigation, it is easy to identify infected media, but very difficult to ascertain the order in which they became infected ∎

Virus Prevalence Table - December 1993

Virus	Incidents	(%) Reports
Form	17	40.5%
New Zealand 2	9	21.4%
Flip	2	4.8%
Keypress	2	4.8%
Vacsina	2	4.8%
Anti-CMOS	1	2.4%
Cascade	1	2.4%
Form.B	1	2.4%
Jack Ripper	1	2.4%
Jan800	1	2.4%
Liberty	1	2.4%
Simulation	1	2.4%
Spanish Telecom	1	2.4%
Tequila	1	2.4%
V-Sign	1	2.4%
Total	42	100.0%

IBM PC VIRUSES (UPDATE)

The following is a list of updates and amendments to the *Virus Bulletin Table of Known IBM PC Viruses* as of 17 January 1994. Each entry consists of the virus name, its aliases (if any) and the virus type. This is followed by a short description (if available) and a 24-byte hexadecimal search pattern to detect the presence of the virus with a disk utility or a dedicated scanner which contains a user-updatable pattern library.

Type Codes

C	Infects COM files	**M**	Infects Master Boot Sector (Track 0, Head 0, Sector 1)
D	Infects DOS Boot Sector (logical sector 0 on disk)	**N**	Not memory-resident
E	Infects EXE files	**P**	Companion virus
L	Link virus	**R**	Memory-resident after infection

Akuku.889.C — **CN:** An 889-byte virus. Detected with the Akuku pattern.

AVV.1667 — **CN:** The AVV virus seems to have been derived from a variant of Pixel, but is sufficiently different to justify the creation of a separate family. Detected with the Pixel-277 pattern.

Better World.C — **ER:** Very similar to the other two variants, and detected with the Better World (Fellowship) pattern.

BUPT.1279 — **CER:** This variant is detected with the BUPT (Traveller) pattern.

Burger — **CN:** Several new 560-byte variants of this overwriting virus are now known, but they are all detected with the Burger pattern. The new variants are 560.V, 560.W, 560.Y, 560.AG and 560.AI.

Cascade — **CR:** Several new variants have been found, including the following minor variants, which are detected with the generic search strings for the 1701 and 1704-byte variants: 1701.I, 1701.J, 1701.K, 1704.N, 1704.O, 1704.P, 1704.Q, 1704.R. In addition, five new variants require new search patterns, as the encryption loop has been modified. The fifth new variant, Cascade.1701.H is detected with the same search pattern as the ‘G’ variant.

```
Cascade.1661.B  012E F687 2201 0174 0F8D BF45 01BC 5A06 313D 3125 474C 75F8
Cascade.1701.G  3101 2EF6 872A 0101 740F 8DB7 4D01 B982 0631 3431 0C46 E2F9
Cascade.1704.L  3101 2EF6 872A 0101 9090 8DB7 4D01 BC85 0631 3431 2446 4C75
Cascade.1704.M  2E01 8DB7 4D01 F687 2A01 0174 0CBC 8206 3124 9031 3446 4C75
```

Chaos.G — **CER:** A minor variant, detected with the Chaos (Spyer) pattern.

Dark Avenger — **CER:** Several variants have been reported recently. They are all detected with the Father search pattern, which should be considered a generic pattern for Dark Avenger-related viruses. With the exception of the Uriel variant, these viruses are also detected with the Dark Avenger search pattern. The variants are: Uriel (1200), Jericho (1365), 1800.H, 1800.I, 1800.Rabid.B, 1800.Singapore and 2000.DieYoung.B.

DataCrime_II.1514.D — **CEN:** Detected with the Datacrime_II pattern.

Diamond — **CER:** Six variants have not been listed in *Virus Bulletin* before, but all are detected with the Diamond search pattern. Their names, which indicate their infective length, are: Diamond.485, Diamond.568, Diamond.584, Diamond.609, Diamond.614 and Diamond.978.

DNR — **CR:** The two known variants of DNR, which are 331 and 397 bytes long, are detected with the Black Monday pattern, but this is rather inaccurate. The following pattern will detect only the DNR viruses.

```
DNR             CD21 3DBA DC75 198B 3601 0181 C605 018B 048B 5C02 A300 0189
```

Doteater — **CN:** In addition to the C and E variants which were included in the large archive described in the December edition, two other variants have appeared recently: D, which is detected with the Doteater pattern, and B, which requires a new search pattern. Both are 944 bytes long, like the original.

```
Doteater.B      582E 0158 2EA2 0001 B890 01FF E0B8 9D01 A301 01B8 1625 BAA5
```

Frodo.Frodo.I — **CER:** A new 4096-byte variant. Detected with the Frodo pattern.

Green Caterpillar — **CER:** All Green Caterpillar variants known until now were 1575 bytes long, but this month a 1989-byte variant appeared. The names of the older variants had therefore to be changed: Green_Caterpillar.A now becomes Green_Caterpillar.1575.A and so on. In addition, a new 1575 byte variant, 1575.F has been discovered, which is detected with the Green Caterpillar (formerly 1575) pattern.

```
GreenCat.1989   0E1F 0706 BF00 01BE 3101 B90C 00F2 A406 1FB8 0001 5033 C0CB
```

HLLO The HLLO family contains several overwriting viruses, which are typically written in either C or Pascal. No search strings will be provided for these viruses, due to the high chance of false positives. It should be noted that, just like other overwriting viruses, these are extremely unlikely to spread. The following viruses have been added to this family: HLLO.Ondra, HLLO.Cvirus 2.0 (4829), HLLO.4372, HLLO.4340, HLLO.3521, HLLO.4778, HLLO.Harakiri.B (5488), HLLO.3008 and HLLO.4096.

Ionkin **CN:** Two new variants of this Russian virus, 218 and 300 bytes long. The 300-byte variant is detected with the Ionkin pattern.

```
Ionkin.218      3F8D 5602 B903 00CD 2173 03EB 5490 8D5E 028B 1F81 FB4D 5A74
```

Japanese Christmas.600.F **CN:** A minor, 600-byte variant detected with the Japanese Christmas (Christmas in Japan) pattern.

Jerusalem **CER:** Most new Jerusalem variants are still detected with the two main Jerusalem family search strings (Jerusalem-US and Jerusalem.1735). This includes the following: 1765, 1808.Frere.C, 1808.Payday.C, 1808.Payday.D, 1808.Periods, 1808.Sumsdos.AL, 1808.Sumsdos.AM, 1808.URI, Anarkia (1829 bytes) and 2132. The following three new variants are also detected with previously published search patterns: AntiCad.3012.E (Plastique), GP1.1533 (Jer-GP1) and Sunday.G (Sunday). One new variant, Mummy.2.1.B, requires the following search pattern.

```
Mummy.2.1.B     2638 05E0 F98B D783 C203 B800 4B06 1F0E 07BB 3E04 9C2E FF1E
```

MG **CR:** Two new 500-byte variants which can be detected with previously published patterns: 2.C (MG) and 5.B (MG-3).

MGTU **CN:** Two new variants, 273.D and 269, detected with the MGTU pattern. The 269-byte variant is actually a corrupted 273-byte variant: one byte has been overwritten, and a B4h changed to CCh, so a block of 4 bytes is not written to infected files.

Multi.B **CER:** Same size as the original version (2560 bytes) and detected with the Multi pattern.

Murphy **CER:** Only two new Murphy variants have been reported recently, Murphy.Swami.C and Murphy.Swami.D, both 1250 bytes long. The C version is detected with the HIV pattern.

```
Murphy.Swami.D  0306 0600 83D2 0029 D819 CA72 0429 0606 0089 FE33 FF0E 1F81
```

PS-MPC This month's crop of PS-MPC-generated viruses: G2.341 (CN), 344 (CN), 346 (CN), 348 (CN), 361 (CN), G2.425 (EN), G2.429 (CN), G2.438 (ER), Scrunch.458 (CN), Deranged.490 (EN), 565 (CEN), 569 (EN), 572 (CEM), 573.A (CER), 573.B (CER), 577.A (CEN), 577.B (EN), 578.A (CEN), 578.B (CEN), 578.C (EN), Viraxe (CN 582), Antiprint (ER 583), 598 (CEN), G2.598 (CER), G2.Sucker (CEN 600), 606 (CEN), Skeleton.626 (EN), Eclypse (CEN 641), Generix (CEN 673), Page.696 (CEN), Z10.763 (EN), 927 (CEN), McWhale.1022 (CER) and 1706 (CEN).

Rage.486 **CN:** A 486-byte variant. Detected with the Rage pattern.

Red_Diavolyata.830.C **CN:** Detected with the Red Diavolyata (MLTI) pattern.

Screen+1.1654 **CER:** This 1654-byte virus is detected with the same search string as the original 948-byte variant.

Seventh Son.426 **CN:** This new, 426-byte variant is detected with the Seventh Son pattern.

Suriv 1 **CR:** Two new Suriv 1 variants, which can be detected with old patterns: April_1.D (Suriv 1) and Xuxa.1405 (Xuxa). In addtion, a new 874-byte variant, named Pizzolla, requires a new search pattern.

```
Pizzolla        0E1F B940 00B0 2EF2 AEB9 0300 BE11 04F3 A674 06E9 6D01 E95D
```

Suriv 2.B **ER:** A previously unlisted variant of the Suriv 2 virus. Not significantly different from the original, and detected with the same pattern.

Syslock **CER:** Two Syslock variants, detected with existing search strings. They are Syslock.C (detected with the Syslock.D pattern) and Syslock.E (detected with the Macho pattern).

Vcomm.633 **EN:** A 633-byte variant, detected with the Vcomm pattern.

VCL New VCL-generated viruses include four companion viruses: Poisoning (706), Succubus (776), Teknitov (969) and Earthquake (1145), four overwriting viruses (347, 409, 429 and 527 bytes long), and the following non-resident, COM-appending infectors, most of which are encrypted: 380, VF93 (380), 445, 506, 507, Azrl.549, 573, 604, Azrl.606, Vpt (606), 610, Anti-gif (628), ByeBye (698), Red Team (716), Elena (730), Annoyer (756), Mexican (786), 951 and Dragon (1005).

VCS **CN:** Two new VCS-generated viruses have been reported. The Dr No variant is detected with the VCS 1.0 pattern, but the Sleeper variant has been modified, so a new pattern is required. As a result of the modifications the virus does not function properly, and second generation samples will probably crash.

```
VCS.Sleeper     BC20 01B9 0F04 8BF7 AC32 C4AA E2FA C390 5E81 EE03 0156 E8E2
```

Vienna

CN: The number of new Vienna variants is not as great as the number of PS-MPC-generated viruses, but is nevertheless quite high. The following Vienna.648 variants are detected with the GhostBalls and Vienna.1239 patterns (those patterns should be considered ‘generic’ Vienna search patterns): AE, J, M, Q, R, S, Lisbon.F, Reboot.B, Reboot.C, Reboot.H and Reboot.G. Eleven more variants are detected with ‘old’ patterns: 353.B (Vienna-5), 573 (Vienna-1), 648.Abacus and 648.Lisbon.G (Dr. Q), 582.B and 583.C (Interceptor), IWG (Violator-B2), 670 and 758 (Violator), 648.X (Vienna-2) and NTKC.B (Dr. Q, Interceptor and 623.B). Finally, there are five more variants which are not detected with the Vienna patterns already published.

```
Vienna.648.H    ACB9 0080 F2AE B904 00AC AE75 EDE2 FA5E 0789 BC16 0089 F781
Vienna.435.B    8E1E 2C00 AC3C 3B74 093C 0074 03AA 75F4 33F6 1F89 768A 807D
Vienna.566      8E1E 2C00 8BFB 83C7 1926 8B77 04AC 3C3B 7409 0AC0 7403 AAEB
Vienna.IT.454   ACB9 0080 F2AE B904 00AC AE75 EFE2 FA07 89BE F402 8DBE 2703
Vienna.833      5153 50BE ???? 0E1F 8A44 FF89 F381 EB51 02B9 8B01 3007 43E2
```

Virdem.1336

CN: Three variants of the Virdem virus (Bustard.A, Bustard.B and Cheater), are all detected with the Virdem pattern.

Youth.580

CR: A 580-byte variant. Detected with the Youth pattern.

ERRATUM:

The pattern used for the CyberRiot virus (*VB*, January 1994, p.13) is incorrect. The following pattern should be used:

```
CyberRiot       B40D CD21 0E07 8B5E F8B9 8000 518A D1B9 FF00 518A E9B8 0203
```

VIRUS ANALYSIS 1

Jan800 - Cause for Concern

Jim Bates

The large majority of viruses which come my way are tediously familiar, incompetent lumps of code which succeed only in confirming the ineptitude of the people who write them. Occasionally, however, there is one which provides a challenge - usually in trying to deduce what the author was trying to design.

The virus described here does not really fall into either category. It *is* boringly familiar, but not incompetent. It contains no illuminating messages or clever tricks, nor does it attempt self-concealment or Machiavellian corruption. In fact, the very straightforwardness of the code has a sinister feel all of its own. Examination of this virus is a little like meeting a brutal serial killer who seems to be a perfectly sane - even likeable - individual. For all its simplicity, I find this virus one of the most worrying I have yet examined!

Jan800, and is a memory-resident, parasitic virus which infects both COM and EXE files by appending its code to them. It contains a destructive trigger routine which overwrites part of the disk, operating one hour after the virus is first loaded on the first of January (any year).

Installation

Each time the virus is executed, it calculates its own location in memory, and sets an index register to a value which enables it to access its own local data areas. Once this index is set, the code accesses low memory and checks the value of the first byte of the Int 83h interrupt vector address. This interrupt is normally reserved for use by BASIC, and therefore may cause some problems on machines which use the language. If this value is 86 (i.e. the letter ‘V’), the code assumes that the virus is resident, and processing passes back to the host program.

If the virus is not resident, processing passes to the installation routine. This checks whether the current program is running in the last Memory Control Block. If it is not, the virus exits to the host program. Otherwise, the virus steals 832 bytes from the top of the MCB, copies all of its code (800 bytes) into this area and passes control to it.

Once resident in high memory, the code collects the top 16 bits of the main (32-bit) system timer and stores them for future reference. The relevant interception routine is then hooked into the DOS services function at Int 21h and the ‘V’ marker is loaded into the Int 83h vector. Finally, the virus passes control back to the host program, leaving itself resident and active in the system.

Infection

The interception routine examines each call for the DOS services and intercepts the following requests: Open_File (function 3D, subfunctions 00h and 02h), Get/Set_File_Attributes (function 43h) and Load_and_Execute (function 4B00h)

The action taken during each intercepted call is the same, and the processing proceeds as follows. First, the caller’s registers are saved on the stack and a flag is set to indicate that the virus is busy. Then the filename associated with the intercepted call is examined to determine its extension. If this is neither EXE nor COM, control returns to the DOS services; otherwise, the virus makes further checks for files matching the names CLEAN???.EXE or SCAN????.EXE.

If the target filename matches either pattern, the original DOS call is immediately allowed to proceed. Note that while this does prevent infection of these files, it does not prevent the virus operating when such programs are being executed. Should the target file be deemed suitable, the virus installs a dummy Int 24h handler to avoid any disk errors being reported to the user. Once this has been carried out, processing collects the file attributes and checks to see if the infection target is marked as a system file.

If this is the case, the file is rejected, and the original call is allowed to proceed; if not, the collected attributes are pushed onto the stack and the file is reset to allow modification. The date and time of the file are then also collected and stored in a similar fashion. Next, the first two bytes of the file are read into memory and checked for the 'MZ' marker which identifies EXE files. If it is present, a type flag is set.

The virus then reads a further chunk of the target file (depending upon the type flag) and tests the collected data for a word value of 5A4Dh (the same as the EXE 'MZ' marker) at a file offset of 07h for COM files and 12h (the checksum field) for EXE files. The presence of this value indicates that the file is already infected and causes the virus to abort the infection process.

"This is the first time I have seen this technique used: it is another indication that the author is not one of the usual adolescents"

Depending on the file type (as determined by a flag set earlier), processing now branches. For COM files, the actual size of the file is checked and infection only takes place if the size is equal to or less than 60,000 bytes in length. For EXE files, the MinAlloc field is checked for a value greater than 14h. The MinAlloc field within the EXE header is used to indicate to DOS just what the minimum memory requirements of the program are. This is the first time I have seen this technique used: it is another indication that the author is not one of the usual adolescents.

If the target EXE file is found suitable, the virus completes various calculations to rationalise the image and file sizes before rejoining the COM infection routine. At this point, the virus code is appended to the target file and the header (24 bytes for EXE files, 8 bytes for COM files) is rewritten.

Finally, the virus repairs the Int 24h vector and completes a check for the trigger conditions before returning to DOS to complete the original request.

The Trigger Routine

When the virus is first executed, the high word value of the system timer is collected from low memory and stored in the virus data area. This is used to determine a delay factor of approximately one hour from initial installation.

Once this delay has passed, the virus instigates another check on the system date. If this is the first of January (any year), the trigger routine is activated. This consists of accessing the current default drive and writing garbage code over its first eight sectors.

The actual routine uses the Absolute Disk Write function (Int 26h) to achieve this, and thus the 8 sectors in question will be logical sectors 1 - 9 inclusive. This will corrupt the first part of the first File Allocation Table on the relevant drive. Recovery may only be possible with expert help.

Conclusions

This virus contains none of the histrionic rantings which typify most other viruses I have seen, and goes about its distasteful tasks in a disturbingly quiet and unostentatious manner, which I find strangely unsettling.

Nevertheless, as it stands it represents no real threat to the computing community at large. It is not encrypted and so will be easily recognisable by even the simplest of scanners. Its trigger routine is reasonably highly targeted, making it unlikely to be a major hazard, and once again is of interest only for its nuisance value.

Jan800

Aliases:	None known.
Type:	Resident, parasitic appending (800 bytes long).
Infection:	COM and EXE files.
Self-recognition in Files:	'MZ' at offset 7 of COM files and offset 18 of EXE files.
Self-recognition in Memory:	56h ('V') at address 0000:020Ch.
Hex Pattern (in file and in memory):	`8E50 81EE 0301 33C0 8ED8 803E 0C02 560E 1F75 3DFC 2EF6 84FC`
Intercepts:	Int 21h, functions 3D00h and 3D02h (Open_File), function 43h (Get/Set_Attributes), and function 4B00h (Load_and_Execute).
Trigger:	Approximately one hour after initial installation on January 1st (any year) the virus will overwrite logical sectors 1 to 9 inclusive of the current default drive.
Removal:	Disinfection of system and files is possible under clean conditions. Recovery of machines affected by trigger routine may be possible.

VIRUS ANALYSIS 2

3NOP: A Looking-Glass War

Eugene Kaspersky

Regular readers of *Virus Bulletin* should remember the article about the 8888 virus, the ‘poor man’s Commander Bomber’ (August 1993, pp. 12-13). This virus is a parasitic file infector which appends itself to files in the standard manner. However, it occasionally calls a routine which overwrites the middle of the host file with virus code, making no change to the start of the infected file.

Although the host file is inevitably corrupted after such an infection, it is often capable of spreading the virus further when it is run. As the start of the file does not point to the virus code, it is necessary to scan every file byte by byte, to ensure that it is not infected. This is the main feature of 8888. However, 3NOP, a new virus which takes this technique one step further, has been discovered in the wild.

A New Type of Multipartism

At first sight, 3NOP appears to be merely another 512-byte variant of the Stoned virus, an impression which holds true for the first half of the analysis. Most of the virus code is fairly standard: once executed, it hooks Int 13h, and infects floppy disks and the Master Boot Sector (MBS) of the hard disk. The remainder of the code comprises a routine designed to infect files.

The main function of 3NOP is to infect the boot sector, although it has a trigger routine which overwrites randomly-selected sectors of the disk with the virus code. 8888, on the other hand, is only capable of infecting files.

In some cases, files infected by both 3NOP and 8888 have had no alteration made to either the file header, the file end, or the length of the file. Each virus is capable of replicating from such files, albeit in a somewhat erratic way, and both pose a similar problem for scanner developers. 3NOP appears to be, in the dirty looking-glass of the computer underground, almost the mirror image of 8888.

Boot Sector Infection

On loading from an infected disk, 3NOP reduces system memory by 1K, by decreasing the word at offset 0000:0413h. It then copies itself into this free memory, after which it hooks Int 13h.

If the system was booted from an infected floppy disk, the virus checks if the Master Boot Sector of the hard disk begins with two NOP instructions (9090h). Should these instructions be present, the virus assumes that the disk is already infected, and the infection routine aborts. If this is not the case, the virus stores the original boot sector in the second physical sector before writing itself into the boot sector location. Control subsequently passes to the code in the original MBS.

The virus intercepts only two functions of Int 13h - Read_Sectors and Write_Sectors (AH = 02h and AH = 03h). These are used for stealth and infection.

On reading from the first cylinder of the floppy drive, the virus infects the boot sector of the disk. This operation is carried out every time a new floppy disk is accessed. Therefore, if the user inserts a new floppy disk into the A: drive of an infected machine, and types ‘A: <ENTER>’, the disk will be infected immediately.

As with the hard disk, the virus checks the beginning of the diskette’s boot sector for the presence of two NOP instructions, which would indicate that the disk was already infected. If this condition is not satisfied, the virus then reads the parameters of the floppy disk, calculates the position of the last cylinder, and writes the original boot sector into it.

“it is impossible to repair the disk simply by using the command FDISK /MBR”

This code is rather carelessly written; in some cases, damage is caused to the data stored on the floppy. The virus does not save the floppy disk’s BIOS Parameter Block (the disk structure information which is stored at the sector beginning). This can confuse the system, and may lead to such problems as inaccessibility of floppy disks.

Partition Perils

The virus uses elementary stealth techniques in order to hide its code from prying eyes. When 3NOP intercepts an Int 13h call to read the MBS of the hard disk, it checks that sector for infection. If this is the case, the virus loads and returns the original contents of the MBS. The virus does not carry out any such redirection when accessing the infected boot sector of a floppy disk.

This obviates the need for the virus to keep a copy of the partition information in the MBS, as all calls to the MBS will return its original contents. However, this means that the hard drive will not be accessible if the machine is booted from a clean floppy disk, as DOS requires the partition information in order to ascertain the layout of the drive.

In addition, it is impossible to repair the disk simply by using the command FDISK /MBR. On floppy disks there is no such stealth: in these cases, disinfection can be achieved simply by using the SYS or FORMAT commands.

File Infection

The virus uses the Int 13h function 03h (Write_Sector) to corrupt/infect files. If the virus intercepts an Int 13h call to write two or more sectors to disk, under certain circumstances it will insert its own code into the file, making 3NOP a rather crude multi-partite virus.

The virus checks the first three bytes of the data being written to disk for a JMP NEAR assembler instruction (opcode E9h). If this first instruction is a JMP to an offset greater than 5000h bytes, the virus overwrites 200h bytes of the write buffer with its own code. When the Int 13h call is allowed to finish, the contents of the modified write buffer are written to the disk.

Any file infected by this method will be permanently corrupted, because the virus does not save data which has been overwritten. When an infected file is executed, the MBS of the hard drive is infected before the host file is unceremoniously terminated, and control passed to DOS.

The code placed in the Int 13h write buffer is different from that contained within an infected boot sector, as the virus uses a different installation routine when running from an infected file. The boot sector part of the virus begins with two NOP instructions, whereas the virus begins with a JMP instruction to an installation routine used when being run from a file. This is a jump to within the virus code, and thus has an offset value of less than 5000h. This prevents the virus overwriting itself, though it may be present in several different locations in a file (see below).

Implications

Infected files are capable only of spreading the virus, and will not work as they did before infection. In some cases, however, the virus writes itself into the middle of a file: if the size of the file concerned is fairly large, DOS saves it not as a single block of data, but sector by sector. Therefore, there are several Int 13h calls, each of which can cause the virus to infect the file at a different location.

If the Int 13h Write_Sector call saves sectors into the middle of the file, and the first sector of the data buffer contains a JMP opcode, the virus can overwrite that particular sector with itself, and insert itself into the middle of a file. That file would then also be corrupted, but could in theory spread the virus as efficiently as a file which is infected at the beginning: the virus' hard disk infection routine does not depend on the offset of the virus code within the file.

However, not all files infected in such a manner will run successfully. The host file may jump to an arbitrary location within the virus code, or the start of the virus code may not be aligned with an instruction boundary. In either of these cases, the most likely result would be that the PC crashes.

An infected file does not necessarily start with a JMP instruction pointing to the virus code: although the virus itself always starts with a JMP instruction, the code could be located at any sector beginning within an infected file. In theory, a file infected in this manner could work correctly until a particular branch instruction occurs, whereupon control is passed to the virus. Of course, the probability of this happening is small. In many cases, infected files will simply cause the computer to hang.

However, the possibility of the virus replicating in this manner exists, and therefore the problems raised by it need to be addressed by the anti-virus industry. Files infected with 3NOP may have the virus code placed anywhere within them, making it necessary for a scanner to examine the entire file byte by byte. This will obviously have a performance impact on many products.

Final Notes

Fortunately, when infecting the hard disk by execution of a corrupted file, the installation routine of the virus checks the DOS address of Int 13h handler for specified values. This means that infected files will only replicate on certain versions of the BIOS. Moreover, the virus has no trigger routine, although it contains about 40 NOP instructions at the end of the boot sector, which is sufficient space for either a trigger routine, or a more sophisticated infection algorithm.

Although the length of the standard hard drive sector is very short (only 512 bytes long), it is nevertheless once again possible to see that brevity of this sort is more than sufficient to encode quite artful methods of infection.

3NOP

Aliases: None known.

Type: Memory-resident, multi-partite.

Self-recognition on Disk:

Checks first two bytes of sector for 9090h (NOP, NOP instruction).

Self-recognition in Memory:

None.

Self-recognition in Files:

See analysis.

Hex Pattern:

```
FA33 C08E D88E D0BC 007C 8BF4
FBA1 1304 48A3 1304 B106 D3E0
```

Intercepts: Int 13h for infection and stealth.

Trigger: No trigger routine, but infected files are permanently corrupted.

Removal: Specific and generic removal possible under clean system conditions. Infected files should be deleted. See analysis for further information.

VIRUS ANALYSIS 3

Lamers Surprise

A recent report from a site in the UK Midlands reveals a new virus in the wild, seemingly indicating the emergence of another virus writer. The virus, which appears completely new, is called Lamers Surprise v1.00. ‘Lamer’ is hacking slang for a fool or a blockhead: in this instance, it represents a barely adequate description of the writer.

The virus is non-resident and parasitic, infecting EXE files by appending 1318 bytes of code to a file, and modifying the EXE header to ensure that its code is executed first. The programming is appallingly poor, and serious errors in the code will prevent it penetrating infected machines. Its principal aim appears to be to hinder disassembly by using an index register to refer to data, forcing the disassembler to maintain details of ongoing register values to classify various data storage areas. This causes only minor inconvenience during disassembly and analysis.

Operation

When an infected file is loaded, the virus code is executed. The system transfer buffer address is then collected and stored, and reset to point into the virus area. After initialisation of various data areas, processing collects and stores the current directory name of the default disk drive. The code then searches for EXE files, using DOS functions 4Eh (Find_First) and 4Fh (Find_Next), including those with Read Only and/or Hidden attributes.

Once such a file is found, its time field is checked for a value of 60 seconds (the virus self-identification marker). Files already infected are ignored. Next, a check is made of file size: those longer than 524,287 bytes (07FFFFh) are rejected. Another check appears to be an attempt to set a minimum size for infection, rejecting files of less than 17 bytes, as well as those with a range of intermediate sizes.

If no suitable files are found, the virus checks the value of an internal counter, which is set to zero when an infected file is first run. The virus then changes directory to the root of the default drive and attempts to locate another suitable file. If it fails, the counter is checked again. This time the value equals one, so processing branches to another routine, which seeks an available subdirectory off the root. When this is found, a variable loop sequence counts subdirectories until either the loop count is exhausted, or no more subdirectories are found. The virus then logs into the found subdirectory and seeks a suitable EXE file.

Once such a file is found, processing passes to the infection section and thence to the exit routine. Thus, only one file is infected on each invocation of virus code, but the location of such files is impossible to predict. If the search for a suitable file fails, processing passes to the exit routine, which is designed to hand control back to the host program. However, several bugs prevent the process happening properly. Effects are difficult to quantify, but tests indicate that there is a 10-15% chance (more if the host program is *Windows*-specific) that the host program will not run properly.

Infection and Trigger

The apparent intention of the virus writer was to infect EXE files by appending virus code and modifying the original file header to ensure virus processing. However, the infection routine will often result in a corrupted header, with unpredictable effects when a file is executed.

There is no trigger routine in this virus, although there are some plain text messages contained within the code. These include the following kind thoughts:

```
 — Oh No!, All your files are as good as dead,
Data files manipulated (only slightly) and
executable infected with a BIG FAT VIRUS.
—YOU FUCKING LAMER—
```

However, there is no evidence, either in the code or from tests, indicating that this virus deliberately affects data files. Similarly, the assertion, ‘All your files are as good as dead’ is incorrect - most infected files can be easily disinfected.

This virus represents no real threat to users. It is riddled with errors, and detectable by even the poorest scanner or integrity checker. As only one report has been received, the writer might be traced and identified. If so, he may find that writing a virus is not as clever as he originally thought.

Lamers Surprise

Aliases:	None known.
Type:	Non-resident, parasitic.
Infection:	EXE files with standard ‘MZ’ header.
Self-recognition in Memory:	None necessary.
Self-recognition in Files:	60 in seconds field of directory entry.
Hex Pattern:	`0344 068B 5408 038C 9200 8B9C 9400 83EE 3383 EE20 8944 0389`
Intercepts:	None.
Trigger:	None.
Removal:	Disinfection possible with specific information on internal virus structure.

INSIGHT

Fighting Fire with Fire

Megan Palfrey

In 1989, Joe Wells encountered his first virus: Jerusalem. Wells disassembled the virus, and from that moment onward, has been intrigued by the properties of these small pieces of self-replicating code. In less than five years from this first incident, Wells has become an expert on computer viruses, and is now partly responsible for the development of one of the best-known anti-virus products, *NAV 3.0*.

Genesis

Wells' first brush with computer viruses did not immediately take him into a career in the data security industry. After leaving his current job, he spent eighteen months working as research editor at a business magazine. During that time, his interest in viruses remained a hobby - but for the fact that he started to review anti-virus products, this might have remained the case.

Unsurprisingly, when writing reviews, his experiences with vendors ranged from one extreme to the other. He recalled two companies sending him their 'latest' viruses along with their 'latest' scanners. Feeling obliged to be fair to other anti-virus companies, Wells sent the viruses he had received to two other vendors, to allow them to amend their products. Although they gratefully accepted the viruses, they felt themselves ethically bound not to release their own libraries, even if it meant a better score in a review of their product.

Even at those early stages, the ethics of 'distributing' viruses was not confined to the anti-virus industry. Wells remembers a cry for help from a woman who asked him to examine her computer after she had had a disagreement with the 'consultant' whom she had employed to set up her system. 'He insisted on teaching her *WordStar*, but she wanted *WordPerfect.*' recollects Wells. 'She fired him, but he returned once to "finish a setup". He rebooted her machine from a floppy disk and left. Two days later, she was greeted by a message that Disk Killer was "processing" her drive. I gathered evidence for her, but she was afraid to pursue it. It seemed that other "virus threats" were also involved.'

A Growing Problem

It was not long after this that *Certus* and *Microcom*, the companies which had not passed on any virus samples, approached Wells with job offers. Although he accepted *Certus'* offer, his contact from *Microcom*, Glenn Jordan, became his closest friend and ally in the industry.

Wells started at *Certus* in 1991, as a 'Virus Specialist'. Soon after his arrival there, it was decided that a new product was needed, which would meet the demands of a burgeoning problem: thus, *Novi* was conceived. His involvement with the product concerned virus-specific detection, file repair, and information systems. Even at this stage, he was more heavily involved with the research side than with programming.

"Less than one percent of homes burn down, but I would recommend that all homes have a smoke detector"

Among the many tasks Wells was assigned at *Certus*, he was asked to develop alliances within the anti-virus industry. This led to him becoming involved with Ken van Wyk's ad hoc group, which cooperated in disseminating anti-virus knowledge. In 1992, they amalgamated with *CARO* (*Computer Anti-Virus Research Organisation*).

Wells believes that the *CARO* cooperative is one of the most useful in the industry: 'My relationship with *CARO* has proven symbiotic, and is quite satisfying, due to my fact-processing addiction. Its strength lies in the fact that, like a good marriage or friendship, participants are there for what they can add, rather than what they can get out of it. It is founded entirely on trust.'

Mix and Match

Certus was acquired by *Symantec* in late 1992, and many of the techniques which had been developed for *Novi* went into *Norton Anti-Virus* (*NAV*) . The addition of the *Peter Norton Group's* utility libraries, as well as the availability of a staff of programmers and quality assurance personnel beyond the means of *Certus*, greatly enhanced the systems being developed for *Novi*. Wells was heavily involved in the development of *NAV3.0*, which (under the name of 'virus sensor') has *Novi's* file watch built into it. The heart of the *NAV 3.0*'s main scanning engine also has a *Novi* pedigree: it is an enhancement of *Novi's* 'warp drive'.

As with *Novi*, Wells' responsibilities involved virus-specific systems. The basic design of the detection, repair, and information systems is his, although shaped and enhanced by the work of many other programmers.

Symantec's interest in the anti-virus market is hardly surprising, according to Wells. He believes that the anti-virus market is growing in proportion to the virus problem, and as the computer universe accepts viruses more as a fact of life. Many companies, he says, are already budgeting for multiple anti-virus products: 'Most people with whom I deal already have more than one anti-virus product, as they have more than one editor, more than one backup, etc. As this becomes the norm, anti-virus product concordance becomes more of an issue. Fortunately, nearly all anti-virus product developers (except Central Point) have accepted responsibility for keeping their product compatible with others.'

Although *Symantec* is a far bigger organisation than *Certus*, Wells is still very much in touch with the needs and problems of the user as well as technical developments. Wells' job description at *Symantec*, according to his manager Jimmy Kuo, is 'walking virus encyclopaedia'. A large part of his job ('Happily!' says Wells) still consists of answering virus questions and helping users.

Views on Viruses

'I tend to lean more towards research than development,' said Wells, 'but only if the research accomplishes something useful. I once heard knowledge likened to a pool of water. Without constant input it stagnates, and without someone using it, it is wasted. As a research editor and virus researcher, my work seems always to revolve around collecting, analysing, coordinating, collating, and releasing information for others to use.'

He feels that viruses are still less than a 'one percent problem': less than one percent of known viruses are common and are on less than one percent of machines - 'this is not to say that the problem is not critical. Less than one percent of homes burn down, but I would recommend that all homes have a smoke detector,' says Wells. Although the glut of new viruses is quite out of hand, the number in the wild is still just over 100 and therefore, he feels, eminently controllable. Most anti-virus reviewers today are 'stuck in the scan age', according to Wells, and have no concept of how to test and review integrity systems. 'So, they keep feeding their readers the lie that detection rate is everything.'

He sees prevention as being more effective than cure: when Wells receives a new virus, he infects the system to see what it does, then uses *NAV's* inoculation system to detect and repair all the infections. This, in his view, is quick, easy, and effective. He believes that integrity systems will be the way forward into the next century, although an anti-virus product should at the very least know all currently in-the-wild viruses. It should be able to clean up a system, and then install a good integrity management system.

'After installation,' he observed, 'the combination real-time and interactive integrity systems can handle the new viruses that appear.' He believes that anti-virus products will develop along both generic and specific lines, but with virus-specific detection being crucial only for installing a more intelligent system. Wells views detection of rarer viruses as less important, and able to be done generically: 'We recently received 151 new viruses from a researcher, and detected 150 of them with a current "fuzzy" signature.'

Education is a useful medium in the fight, but although it helps users deal intelligently with viruses, it is not the whole solution: 'Education may limit the number of virus disasters, but not the number of incidents,' says Wells. 'It should be focused towards preparing users, dispelling myths, and maybe teaching computer ethics.'

Personal Points

Wells plans to continue in virus research, and hopes to expand his informational role in the field, by pursuing more projects such as the 'In the Wild' and 'Frequency' lists which he currently collates. He believes that misinformation and bad advice is still widespread: 'Just yesterday I saw a horrifying post on *CompuServe.* A virus "expert" was telling a user to use FDISK /MBR to remove Monkey, which would leave the disk with scrambled partition information. The same trick is often suggested to remove Form, which doesn't infect the MBR at all!'

Although he is vehement in his belief that viruses are a problem which must be controlled by any and every means possible, he also feels that the writing and perpetration of viruses is an ethical issue, not a legal one; therefore, he does not view virus writing as a crime. However, he would probably support legislation about virus programming and virus damages.

"products will develop along both generic and specific lines, with virus-specific detection being crucial only for installing a more intelligent system"

'Even if such legislation failed to pass,' he said, 'at least it would succeed in raising a fact-based awareness of the virus problem. The ERA [Equal Rights Amendment] failed to pass in the USA, but the discussions surrounding it did much to increase public knowledge and change attitudes about real problems.'

'All in all,' said Wells, 'the perspective I've developed in my career is perhaps a bit odd. As a "techie", I still use DEBUG more than any other tool, but when I read a review that rates a product highly because of the interface, the editor in me has to nod in agreement. For *MIS* people who have *Windows* on 80% of their systems and viruses on few, compatibility and usability are the dominant prerequisites. That perspective has been acquired both from the views of a small company, trying to survive, and from a huge corporation, trying to thrive - two very different vantage points. I liked the family feel of a small company like *Certus,* and I miss that. But, despite the corporate atmosphere, I prefer the reach of a company as large as *Symantec*, simply because the information I process now can benefit far more people.'

Wells fights fire with fire, bringing his expertise and the wealth of his experience to bear upon the problems of the user. Can he continue to do so? 'Yes,' he promised.

FEATURE

Morality versus Legality

Simon Halberstam

It seems preposterous that virus authors should have any rights whatsoever over their creations. However, it is not unheard of for a virus writer to threaten to sue *Virus Bulletin* for publishing information on their handiwork, arguing that they owned copyright in the virus code.

This begs the question as to who is actually in the wrong: the virus author for writing the virus, and having contravened accepted computer ethics, or the scanner manufacturer, for using virus code belonging to the virus author in a program essentially designed to seek and destroy the virus. Can the morally wrong be legally right?

The fact that virus authors have any legal rights whatsoever in their work is due to the *Copyright, Designs and Patents Act 1988*, which states, 'the author is the first owner of copyright'. It is thus conceivable that a virus author could sue the author of a scanner program, which includes part or the whole of the virus code, for breach of copyright. The virus author would have to identify himself in such a case, risking prosecution under the *Computer Misuse Act*. Should he already have been the subject of action taken under this act, however, he may feel that he has little left to lose. Does the law provide any protection for the scanner manufacturer?

Comfort for the Righteous

If the virus author is based in a country which is not a party to the *International Copyright Convention*, the UK may not recognise him as the copyright owner, and copyright in the searchstring might then vest in the author of the scanner program. The *EC Database Directive* should also help the authors of searchstring libraries.

This *Directive*, due to be adopted by the *EC Council* in the next few months, is also expected to apply to pre-existing databases. To qualify for protection, a database will have to be the author's own intellectual creation. It will be necessary to establish the originality of the selection or arrangement of the compilation to pass this test.

It would be insufficient to compile an unoriginal set of contents in an original way. If a database is protected by virtue of its originality of arrangement, the rights of the owner will not be infringed if someone copies the contents, but arranges them differently. Thus, if the compiler of a searchstring library can demonstrate originality, he should own a right which he can protect. However, he would have rights only in the compilation as a whole and not in the component parts. In such circumstances, it would only be worth considering suing someone who copies all or a substantial part of the compilation.

Where originality is lacking, there may still be comfort for searchstring library compilers: the *Directive* will probably introduce an 'unfair extraction' right. This will entitle the author of any database to prevent unauthorised extraction or re-utilisation from that database, of its contents, wholly or in part, for commercial purposes. This right will not apply if the contents are already protected by copyright. The exact effect of the right is still a matter of some debate.

"it is conceivable that a virus author could sue the author of a scanner program … for breach of copyright"

Finally, scanner program authors should note that, whereas screen display is not the subject matter of the copyright in a computer program, it may, if original, enjoy a separate copyright which may be infringed and thus also protected.

False Positive?

Even if an author could establish that he owns copyright in a scanner program, his protection is far from complete. From its title, you might expect the *EC Directive* on the 'legal protection of Computer Programs' to be exclusively in favour of the software author. However, in line with the open systems philosophy, the *Directive* (which took effect in the UK at the start of 1993, courtesy of the *Copyright Computer Programs Regulations 1992*) permits an authorised user of a program to decompile the program code, without obtaining authorisation from the copyright owner, in order to create an independent and interoperable program.

Such leeway would be allowed providing the following conditions were met: that information necessary to achieve such interoperability was not previously 'readily available', that reproduction and translation were confined to those program parts necessary to achieve interoperability, and that the information obtained would not be used for development, production or marketing of a program substantially similar in expression to the decompiled program. Unfortunately, some might think, UK regulations do not preclude the use of information for the creation of a program which competes with the decompiled program without being substantially similar in its expression to the original program. We await with interest test cases which interpret the meaning of this.

If the licensor owns copyright in the original program, he could sue, should the competing program be 'substantially similar' to his own. However, it is not difficult to create a competing program which is not substantially similar in its expression to the original program.

The Directive in Practice

The stipulations of the *Directive* have forced many software suppliers to amend their standard forms of licence, to eliminate decompilation bans and other outlawed provisions.

One factor which must be considered is whether or not it is possible to prevent a lawful user of a program from using the decompilation right as a springboard to write and market upgrades and add-ons to a program.

If the program author owns copyright in the program (which, as demonstrated, may be dubious in the case of scanner programs) and the lawful user copies a substantial part of the original program in the upgrade, update, or add-on, then the copyright owner may consider suing. If the program owner does not own the relevant copyright, he may try to prevent a user purchasing upgrades etc from a third party, but might fall foul of competition rules preventing an upstream producer tying customers up in a downstream market.

In view of this, authors of scanner programs who fear the ramifications of decompilation would be well-advised to consider the get-out provided by the *Software Directive* and the UK implementing provisions. This provides that the right to decompile may be contractually excluded where the interface specifications of its programs are made ‘readily available’ to licensees. It is not yet possible to say how this will be interpreted.

Use of the information must be regulated very carefully, as the *Directive's* use restrictions would not apply to information released in this way. It would be advisable (dare I say it!) for anyone who decides to take this approach to seek legal advice in advance.

However, the *Directive* is permissive in respect of error correction by the licensed user, and of other acts necessary to enable the licensee to use the program for its intended purpose. This may well cover a lawful user who wishes to copy a program in order to send it off to the virus lab for a health check.

Avoiding Leaks

Leaking employees (sic) are a large risk for software companies, whether they produce scanner programs or any other software. Employees often pass on valuable information to third parties, knowingly or unknowingly. When an employee or ex-employee copies code in which the employer owns copyright, the employee may be sued for breach of copyright, but it is rare that the situation is so clear cut.

It is impossible to prevent employees leaving with a certain amount of know-how. Nevertheless, there are various legal and practical ways in which an employer can attempt to prevent the confidentiality of his most valuable assets, including searchstrings, being jeopardised by employees both present and past. These measures should systematically address the various stages of the term of employment.

At the start, the incorporation of appropriate and reasonable restrictive covenants into contracts of employment should be considered. Another measure to be recommended is the institution of appropriate company security procedures which limit access to confidential information strictly to those employees who need to have such access for the purposes of the employer's work.

Employers should also take rigorous steps to impress upon employees the confidential nature of the information which they are handling and, in particular, should make employees sign appropriate confidentiality undertakings before they gain access to any confidential information.

Before the door slams shut, an employer should attempt to ensure that a departing employee has returned everything relevant in his possession, whether stored in a home computer or at work, and impress upon him that failure to do so may result in criminal proceedings against him for theft under the *Computer Misuse Act.* Finally, after the departure of an employee, his employer should immediately cancel all his passwords and any user IDs.

The Consequences

Anti-virus companies have an understandable thirst for new knowledge and may be tempted to sign up employees of rival companies who might be able to extend their searchstring libraries. But beware! If the employee was subject to appropriate confidentiality controls by his previous employer, the disclosure of any such confidential information might not only make him liable to his ex-employers but could also make the new employer liable for inducing breach of contract.

These risks will not disappear because the ex-employer has gone into liquidation, as the liquidator will probably defend the company's contractual interests with considerable rigour.

The effect these new laws will have is still not totally clear, and it remains to be seen how policy will be delineated and implemented. Until this happens, no firm statements can be made: any conjectures we might make must stay within the realm of supposition. How much protection is there for the righteous? We must wait and see.

About the Author:

Simon Halberstam trained in the City, holds both English and French law degrees, and specialises in computer and EC law. He previously worked in the fields of IT law and EC law in a large City law firm, and now heads the IT law department of *Alfred P Halberstam*, a firm founded in 1950. Anyone requiring further information on this or any other aspect of IT law may contact him on:

Tel.: +44 (0) 71 405 5382
Fax: +44 (0) 71 404 0919

TUTORIAL

From Little Acorns Mighty Viruses Grow

Alan Glover
Pineapple Software

Although the vast majority of all computer viruses are written for the *IBM PC*, it is not the only computing platform which has been affected - every popular machine has its share of viruses. The *Acorn Archimedes* is no exception, and while the *Archimedes* viruses are nowhere near as advanced as those on PCs, a number of interesting contrasts can be drawn between the two systems.

At present (January 1994) 52 virus families affect the *Archimedes*; a total of 84 viruses if variants within each family are considered. Two commercial anti-virus packages exist (one of which I maintain), and a constantly changing number of PD and Shareware packages are also marketed.

Background

The *Acorn Archimedes* is difficult machine for virus writers. It is much harder for a virus to gain control in an *Acorn* system than on a PC. Both the Operating System (OS) and the windowing environments on the *Archimedes* are held in ROM, making it is impossible for a virus to install itself into those parts of the system which are guaranteed to be re-loaded after a reset. Viruses must therefore get into the system later, as applications are set up.

There is no way to write the equivalent of a Boot Sector virus, as a freshly formatted disk contains only data. Although the current OS automatically detects and accesses PC disks, the Boot Sector of the disk is interrogated rather than executed. Although the most successful PC tactic is unworkable, there are still three common classes of *Acorn* viruses:

!Boot Infectors: A windowing application consists of a directory with an exclamation mark as the first character, the contents of which are partly pre-ordained by the windowing system. One of these files is the !Boot file, which is executed (if present) when a directory viewer containing that application is first seen on the screen. Thus, viruses which infect !Boot files can become active even if the application they have infected has not been used. If the infection algorithm is well-coded, this can be a very effective way of infecting the machine. The !Boot infector is generally the easiest type of virus to detect, since it is conventionally plain text and can be easily parsed by a signature scanner.

Absolute File Infectors: An Absolute is a machine code file assembled to be loaded at a specific location, commonly used for the main program of an application (called !RunImage) to identify it to the windowing environment. Viruses which infect Absolutes are harder to detect, but slower to spread, as they rely on the infected application actually being used.

Relocatable Module Infectors: A Relocatable Module is a machine code file which provides additional commands or facilities available throughout the system. The code must be written in a relocatable fashion, since one memory area is used for all modules. This approach shares some of the weaknesses of Absolute File infectors, but will spread more quickly as certain modules are loaded by more than one application: thus they have more chance of being in memory.

The Beginnings

The first *Acorn* virus came to light in February 1989, at which time I was running a bulletin board. A fellow SysOp circulated a message that a virus had been uploaded, and that he had written a program to remove it and to inoculate files. I managed to obtain a copy of the virus, and called it FF8 (it infected Absolute files, whose filetype number is 0FF8h). That name has since been changed to Archie, to fit in with the message it displays.

In January 1990 Paul Vigay announced that he had written a program called *Virus_Zap*; it later became apparent that he had also written a virus to test the program. The virus was initially called Datadqm after its filename, but was renamed Vigay when virus names were standardised. *Virus_Zap* gradually turned into a program called *Guardian*, the latest incarnation of which is *Inoc3*.

Writing a research virus may be justifiable, but this virus was observed in the wild at least twice (on one of these occasions I suspect it was released through a major BBS). There are now also two new strains created by other authors, so to date, this ‘research’ virus has spawned four viruses in the wild, showing the danger of creating ‘test’ samples.

Towards the end of 1990, a virus calling itself Icon appeared. The original version is unknown; it was popularised in November 1990 when a particularly ill-informed person at a school in Wakefield added his own details to the file to ‘see how far it gets’. The virus is childishly simple to alter (it is written in uncompiled BASIC!), and has the largest number of variants within a family. It has also spawned additional viruses, which either behave significantly differently to the original, or are modelled on its code.

Changes, Developments and Novelties

The first malicious virus, later named Thanatos, was written in late 1990, and marked a change in virus writers’ coding ability. Its messages were rude and often vulgar, and it could damage or destroy data.

During that year, virus spread was generally slow: there were few bulletin boards, and most of these were run by people with the necessary ability to spot suspicious behaviour in an uploaded file. As bulletin boards became more widespread, this changed. The largest *Acorn*-related bulletin board system was *Acorn's SID*, which I took over in April 1990. In March 1991 I was contacted by a disgruntled user who had an infected application, which he thought had originated from the *Acorn* BBS. I reassured him that this was not the case, but the event made me realise that a capable virus removal program was needed to check and vet files on *SID*. *Killer* and *VProtect*, complementary programs specifically tailored to the *Acorn* arena, eventually evolved from this incident.

VProtect is a Relocatable Module which, as well performing as other checks, parses !Boot files before they are executed. *Killer* goes one step further, detecting and removing all viruses known to date. It also has some generic detection facilities, and can disable memory-resident viruses. As the year passed, the need for a virus killer outside *Acorn* became obvious, so *Killer* was made available externally.

Extend, a new virus, appeared in July 1991. It was generally harmless, but spread faster than those previously known. A copy was sent to Richard Lloyd (a student at *Liverpool University*) for analysis: he wrote the program *VKiller* in response. In later versions he expanded *VKiller* to cover other viruses in circulation. Meantime, *Killer/VProtect* were expanding similarly, but were rare outside the *Acorn* arena.

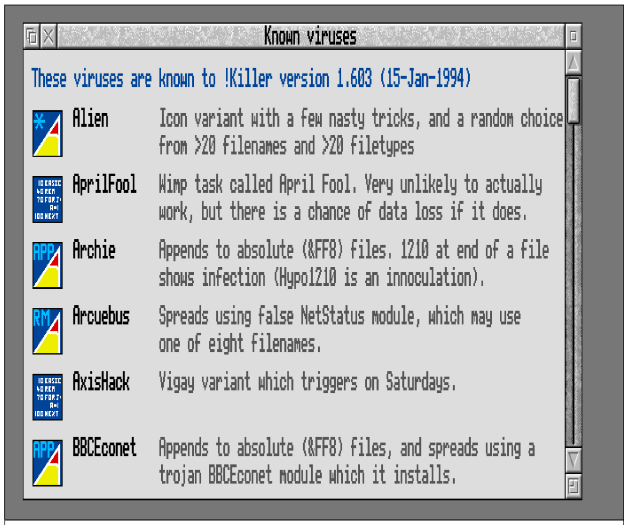
A New Scenario

At the turn of the year another program, and another researcher, appeared. Tor Houghton, who lived in Norway, wrote the program *Scanner* to deal with the viruses he knew: for a time it was the most capable alternative to *Killer/VProtect*. However, his other accomplishment, the conception and authorship of the first *Archimedes Virus Reference Document* (*AVRD*), is even more valuable. It still exists now (albeit much changed), edited by Tor and myself. It is at present shipped with *Killer/VProtect*, and made available on various public BBSs and file servers.

The appearance of the *AVRD* made it obvious that the time had come to standardise on virus nomenclature - at this point, FF8 became Archie, and Datadqm, Vigay.

During that period, it also became apparent that the time needed to maintain *Killer/VProtect* was becoming excessive (the programs were not my only occupations), and it was eventually decided to commission an outside company, *Pineapple Software*, to take over and market the products. Arrangements were made for an annual subscription scheme to commence in May 1992. Despite this change, I retained my involvement with the programs.

During the early months of that year a new virus, called Module, came to light: it made an extremely good job of appending itself to Relocatable Modules, and did much to



These viruses are known to !Killer version 1.603 (15-Jan-1994)

Alien	Icon variant with a few nasty tricks, and a random choice from >20 filenames and >20 filetypes
AprilFool	Wimp task called April Fool. Very unlikely to actually work, but there is a chance of data loss if it does.
Archie	Appends to absolute (&FF8) files. 1210 at end of a file shows infection (Hypo1210 is an innoculation).
Arcuebus	Spreads using false NetStatus module, which may use one of eight filenames.
AxisHack	Vigay variant which triggers on Saturdays.
BBCEconet	Appends to absolute (&FF8) files, and spreads using a trojan BBCEconet module which it installs.

Only a handful of viruses which begin with the letter 'A'! The *Archimedes* has escaped the worst of last year's virus writing mania.

alert users to the dangers of computer viruses. It did nothing to draw attention to itself until 6 September 1992, when it displayed a message. Unlike most previously-known viruses, it created no additional files, so many people missed it. That year also heralded other infections: two magazine cover disks were affected, at least one commercial package, computers on a roadshow, and (worst of all) an EPROM containing driver software for an IDE hard disk. Version 1.26 of *Killer* was made available for through the magazines concerned to ship on the succeeding month's disk. Fortunately, it proved possible to remove the virus cleanly.

Of virus programs available at this stage, Richard Lloyd's *VKiller* was infrequently updated, and soon became obsolete. In 1992 Tor Houghton started studies at *Brighton University*, so updates to *Scanner* have also become rarer (it too is now considered obsolete). However, it was still being produced, as was, sporadically, Vigay's *Guardian*.

As the *Pineapple Virus Protection Scheme* gained momentum, more and more viruses were flushed out. Many had previously gone unreported, as there had been no central body to collate data. Amongst the viruses which came to light in the first few months of the scheme was T2 (thought to have been written by the author of Thanatos): the first, and so far only, virus to cause *Pineapple Software* to perform an immediate update to all users. Even so, we narrowly missed updating everyone before the July 4th trigger went off.

This delay occurred primarily because the person who found the virus chose to take it apart himself before reporting it. Luckily, damage to hard disks was partly reversible, so the effects were to some extent nullified.

This year, the number of both original and modified viruses has continued to rise. However, the surge of activity resulting from flushing out viruses which were undetectable by any software before *Killer* was released has died down. And what will the future bring? I am confident that more viruses will appear, and that they will grow in complexity. Another point to be considered is that many commonplace PC tactics have yet to be employed in *Acorn* viruses.

Of People and Programs

This is a short biography of people who have been involved with the anti-virus scene at some point, and contains notes on programs they have written. I have concentrated solely on central characters, and offer my apologies to anyone who thinks he should have been included. It may seem something of a rogue's gallery, but this is how it really happened! The history of the *Acorn* virus scene is far from clean and decent.

Alan Glover (author): Involvement began 1989 and continues. Co-author and maintainer of the *AVRD*. Author of *Pineapple Software's Killer* and *VProtect*, first written at and for *Acorn*. *Killer* deals with every known virus, and is (in my opinion!) the definitive program available. It is distributed by a subscription (£24.00 per single copy), with site/area licences available.

"My own belief is that the Acorn scene can be regarded as a microcosm of the early days of the PC scene"

Tor Houghton: Involvement began late 1991, and continues, albeit at a reduced level. Tor's Public Domain programs, *Scanner* and *Interferon*, were generally the second most capable, after *Killer/VProtect*, and benefited from access to the information which came into *Pineapple Software*. Although the programs are now obsolete, his lasting contribution is the birth of the *Archimedes Virus Reference Document*, which enabled standardisation amongst *Archimedes* virus researchers. The *AVRD* remains the best source of information about viruses. Like *Virus Bulletin*, it informs, without giving enough detail to aid a potential virus writer.

Richard Lloyd: Involved 1991-1992. For much of this time his *VKiller* was the best anti-virus program which was widely available. However, in mid-1992 support for the program trailed off inexplicably, and he has now all but disappeared from the scene.

Jon Ribbens: One of a group of teenage programmers called *DoggySoft*. His anti-virus PD program, *VEnd*, was released in various Beta versions last year. Since then, little has been heard of it, with further updates unlikely, as Jon has now gone to university. Unfortunately, acrimonious debates on bulletin boards led to a number of people in the *Acorn* field refusing to have anything to do with Jon or *DoggySoft*.

Paul Vigay: Involved late 1989 to present. Author of the Vigay virus, and of an anti-virus program called, at various times, *Virus_Zap*, *Guardian*, *GuardianPRO* and *Inoc3*. Despite regular claims that his programs deal with all known viruses, only a small subset of the viruses which I know seem to be covered. At various times the program has been PD, Shareware, PD again, and is now commercial (forming part of the *Investigator 3* package). Although he has denied writing the Vigay virus, the following quote from the documentation with version 2.10 of *Guardian* (30/1/92) says it all (it also appears in version 3.09 on 14/10/92):

```
Virus Name  Version of !Guardian that deals with it

Datadqm     All versions

This is not a virus as such, due to the fact no actual
harm is done to your disks. It is merely a desktop
silly that is capable of replicating amongst any
application not already having a !Boot file.

This demo was actually written by me, as a short demo
to test early versions of !Guardian. Only four people
were given copies of it, so it shouldn't be too widely
spread. The effect is a screen 'wobble' every Thursday
(Friday on some versions and only Friday 13th on the
final version).
```

While we can argue about semantics, a virus is anything which replicates without permission: Datadqm certainly does that. The screen wobble has caused people to take monitors to dealers, believing them to be faulty. The second paragraph has been deleted in later versions of the software.

Despite the above-quoted comments, two different 'Thursday' versions have been reported in the wild. The virus has been sufficiently widespread to suspect that it was at some point available on a large BBS. The virus continued to evolve after it first escaped. So far, there have been no reports of 'Friday' versions in the wild (although one of the strains upon which it is based does trigger on a Friday).

Conclusion

At the moment, the *Acorn* virus scene is dominated by *Killer/VProtect*, the main contender being the enthusiastically advertised Paul Vigay program. The difference in scale is probably to be expected - the *Acorn* scene is smaller than the PC one. Similarly, activity levels are such that one person can still keep up with all the technical developments.

Newsgroups such as *comp.virus*, and other sources of information about the present state of the PC anti-virus industry, point to the fact that the *Acorn* scene can be regarded as a microcosm of the early days of the PC scene. Techniques are still quite primitive, and many virus writers are undoubtedly bored schoolkids (which gives rise to a notable seasonal pattern in activity levels). The situation is still calm compared to the state of siege some PC users envisage, but serious enough that precautions should be taken wherever a situation which is likely to encourage infection exists.

Users requiring further information on *Acorn* viruses should contact *Pineapple Software*: 39 Brownlea Gardens, Seven Kings, Ilford, IG3 9NL.
Tel +44 (0) 81 599 1476.

PRODUCT REVIEW 1

ViruSafe - True to its Name?

Mark Hamilton

ViruSafe, in its various incarnations, has been reviewed twice before in *Virus Bulletin*. My co-reviewer, Dr Keith Jackson, looked at it in April 1990 (pp.18-19) and I re-viewed it, as *AllSafe*, in January 1992 (pp.19-22).

ViruSafe is produced by the Israeli anti-virus developers *EliaShim*. The product first became widespread when *Xtree* decided to join the anti-virus goldrush and launched *AllSafe*, licensing the code from *EliaShim* (the *Xtree* product was withdrawn in early 1993). Previous reviews of *EliaShim's* scanner have found the product to be reasonable, if slightly lacking in virus detection capability. Has *EliaShim* managed to keep up with its competitors?

The Package

The product is delivered on either a 5.25-inch or 3.5-inch floppy disk (but not both). It comes with a 216-page A5 manual, the large font size of which (it appeared to be about 16 or 18 points) I personally found rather obtrusive. The book does a fair job of describing various functions provided by the software, despite the fact that many of the screen captures used are poorly reproduced.

Another visual distraction is *EliaShim's* insistence on suffixing the name of any product for which a Registered Trademark exists with the symbol '®' every time it appears. In the case of *ViruSafe* [®. *Ed.*], this can be as often as several times per page. Other than these niggles however, the manual tells the user what he needs to know, and therefore serves its purpose.

The software part of the package consists of the following components:

- PCC.EXE, which provides system information like *Norton's* SI.
- PIC.EXE, a generic file checker.
- VSMENU.EXE, the menu-driven front-end.
- VC.EXE, the memory check program.
- VS.EXE, a 16 Kbyte TSR monitor program.
- VREMOVE. EXE, the main scanner and virus removal program (formerly known as UNVIRUS).
- VSCOPY.EXE, which checks for viruses, provided the *ViruSafe* TSR is installed, whilst copying files.

Apart from an icon file, no *Windows*-specific elements are included with this version. I understand that there is a *Windows* version of the software, which contains both DOS and *Windows* versions of the main programs.

Installation

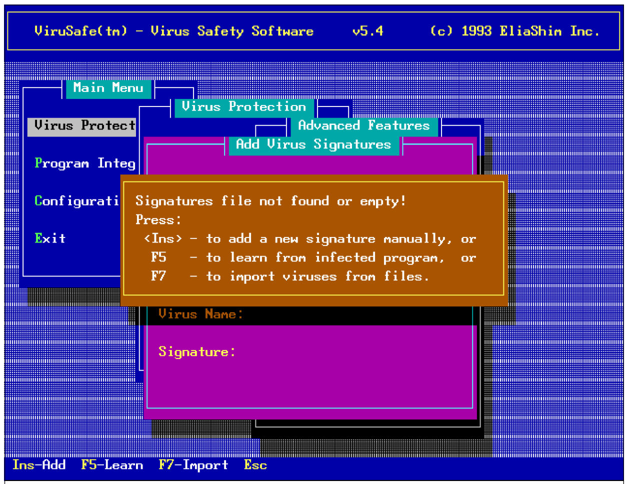
I ran the installation program, which copied the files to my hard disk. *ViruSafe* then requested a registration number, which was to consist of one letter and six digits: it was noted neither on the disk label, in the manual, nor in the letter which accompanied the package.

I attempted to circumvent this request by hitting the enter key, which caused the software to prompt me that 'all fields must be completed'. A call to *EliaShim's* UK distributor was required, whereupon I was informed that any seven charac-ters could be entered for packages bought in the United Kingdom, as copy-protection had been removed from the UK version - a detail which should have been stated somewhere in the documentation. This also raises the worry that *ViruSafe* may be sold in a copy-protected form elsewhere - prospective buyers are advised to check with their local distributor before parting with their money.

The Scanner

EliaShim claims that its scanner, VREMOVE, will scan files contained in a wide range of archive formats, such as those created with ARC, PKZip, LHA, ARJ and so on. Although this is true, *EliaShim* has, by a rather artful dodge, managed to pass compatibility problems into the lap of the user.

When a compressed file is detected, VREMOVE calls a batch file, which in turn calls the appropriate archiving software. This decompresses the archive, which can then be scanned. Sample batch files are provided with the product, but will have to be configured on a machine-by-machine basis. Obviously, this system relies on the fact that the user has the appropriate archiving software on his machine - a reasonable assumption, as compressed files are of little use



```
Signatures file not found or empty!
Press:
 <Ins> - to add a new signature manually, or
  F5   - to learn from infected program,  or
  F7   - to import viruses from files.
Virus Name:

Signature:
```

ViruSafe is capable of automatically extracting a signature from an infected file. This could be of help in the event of a new virus running riot in a system.

without it! This is a simple yet effective technique, and I am surprised that other anti-virus software developers have not yet adopted it.

ViruSafe's handling of dynamically compressed executables was far less successful. When a compression program (such as LZEXE) produces a dynamically compressed executable file, it modifies the executable to include the decompressor and a table used by the decompressor to reconstruct the original executable as it is read into memory. The documentation claims that VREMOVE can scan such dynamically compressed executables: this proved not to be the case.

VREMOVE seems to have a problem decompressing these files reliably. If it fails, it simply hangs the program. I tested VREMOVE on a number of drives, all of which contained dynamically compressed executables (my hard drive contains several examples of commercially obtained programs whose executables are compressed with LZEXE). Every time I attempted to scan these files, VREMOVE hung the computer. *EliaShim* says that no other user has reported problems of this nature, though I found the problem repeatable.

In Operation

ViruSafe floundered somewhat when run against the 'In the Wild' test-set, identifying just 66 out of the 80 samples as infected. This is a disappointing result, placing the product well down the batting order when compared with the other products in last month's Comparative Review (*Virus Bulletin*, January 1994, pp.14-19).

In common with several other products reviewed at that time, it missed Quox and Monkey II boot sector viruses (both of which are known to be in the wild). *ViruSafe* fared better against the 'Standard' test-set in percentage terms, finding 359 of the 371 infections. Importantly, however, it failed miserably in the tortuous 'Mutation Engine (MtE)' test, scoring 1,746 out of 1,926: in real life, one failed detection here could cause severe headaches for users.

VREMOVE provides a number of options controlling what is scanned, how it is scanned, and how results are provided. When scanning a disk, its first action is to scan the boot sector. This is followed by a long pause, before file scanning commences. At first I thought the PC had hung, as the delay lasted for several seconds: I was mistaken, however, and scanning continued. Execution speed seems otherwise acceptable, although users must note that it appears to default to checking only COM and EXE files.

One of VREMOVE's options is to remove some viruses it detects. This requires that the anti-virus program uniquely and absolutely identifies the viruses concerned.

Such an option is one aspect of *any* anti-virus product which fills me with deep suspicion, and one I would not encourage anyone to use. I have said this many times before, and reiterate it here: if your machine is infected with a virus, replace the infected files, either from master disks or from known clean back-up copies.

This point of view was justified when checking the virus reports issued by *ViruSafe*: on more than one occasion, the product identified more than one 'removable' virus in a file known to contain only one infection. How does the software decide which virus removal algorithm to use? The only circumstance in which removal should even be considered is if the virus has been identified precisely - this did not seem to be happening.

Other Components

PCC is an informational utility, along the lines of *Norton's* SYSINFO program, providing detailed information about the computer hardware and system software. This allows the user to display configuration information such as the contents of the computer's environment - for example, the PATH, COMSPEC and PROMPT settings. Other menu options provide an overview of the computer (processor type, memory installed, number and type of I/O ports etc); a memory map for the first megabyte of physical memory used; identification of those adaptor cards fitted in the PC which have an onboard ROM; and full details of the hard disk parameters.

So far, there is nothing uniquely anti- or counter-virus about all this, and certainly nothing not otherwise provided by shareware or by a commercial Systems Information utility. However, *EliaShim* includes a facility for repairing boot sectors called HDFix which, in the company's own words 'provides a last ditch method of repairing damaged Partition Tables and Boot Sectors'. The manual warns that these methods should be used only if all else fails, as the program rebuilds either the Partition Table or the Boot Sector completely from scratch.

In certain cases, the user is prompted to contact *EliaShim* for a code number in order to proceed - why? They may have a perfectly reasonable explanation for this 'safeguard', but I see no reason why a user should need to contact a software company for a special code number, particularly when this is in order to continue with a function which in any case is provided by the software. [*EliaShim explains that users would occasionally try to 'fix' an undamaged disk. In such a case, PCC can do more harm than good. For this reason, certain functions of the program are designed to be used only in conjuction with technical support. Ed.*]

In my previous review two years ago, I mentioned that the VC.EXE component (the memory check program) sets off a mechanism, whilst scanning memory for viruses, which attempts to unearth (unknown) memory-resident viruses. When a new virus is discovered, a copy of its code is saved to VIRUS.PGM. *ViruSafe* has the ability to 'learn' new signatures from such files. This 'learn' mode selects a sixteen-byte signature from the file, which can then be used to scan other files on the disk. Although this sounds like a wonderful idea, and may well work very well for a simple unencrypted virus, the reader should be aware that it will not work when dealing with polymorphic viruses. The facility is a useful addition, not a cure-all.

The user can also enter any hexadecimal signature, such as those published in *Virus Bulletin*. However, these are limited to sixteen bytes, increasing the risk of false positives. Nevertheless, this is a useful tool.

Further Attractions

A new segment of VREMOVE, which *EliaShim* calls 'Correlation', analyses a few bytes from a number of COM and EXE files for similarities: such analysis could provide evidence of infection by an unknown virus. It may also throw up false positives if the programs it compares happen to be created with the same compiler, and therefore contain the same start-up code.

This is exactly what happened when I tried the function on some files which had all been compiled with *Borland* C - but in the 'real world', it is unlikely that many users would have a number of files all compiled in the same way, so *EliaShim's* analysing techniques might well prove beneficial. It should be noted, however, that such a technique will not work with polymorphic viruses.

The generic checker, PIC, works like many others, in that it creates a database containing essential file information against which files are subsequently checked. PIC was capable of detecting subtle one-byte changes made to files. In addition to checking files, PIC will keep copies of the boot sectors and CMOS RAM, and can replace infected boot sectors with original copies or rewrite the CMOS settings. A helpful and effective program.

A useful option buried away in the 'Advanced Features' menu is the ability to create a 'rescue diskette'. If this is done, the diskette will contain copies of the boot sector(s), CMOS RAM information, and a small utility to replace the (infected) boot sectors and rewrite the CMOS. Ideally, such a disk should be made bootable. The authors of the manual seem to have overlooked this prerequisite: such a note would be a nice addition to the documentation.

Conclusions

ViruSafe v5.4 is a moderately effective package - in fact, its mediocrity seems to be its principle drawback, as it has little to recommend it over its competitors. Although the product has an interesting range of generic detection programs, the system must be virus-free at installation: this can only be ensuring by using a good scanner. *User Friendly MicroSystems*, the company's UK distributor, has informed *Virus Bulletin* that *ViruSafe v6* would be available shortly. This will hopefully improve matters.

The problems I experienced when using *ViruSafe* were rather disappointing. The continued crashes when scanning dynamically compressed files, the concerns over the removal mechanism, and the remains of a copy protection mechanism all gave me reason to worry. *EliaShim* must deal with these problems soon: in a market full of competitors, the product is in no way noteworthy.

ViruSafe

Scanning Speed

Hard Disk:

Turbo Mode	unable to complete
Secure Mode	unable to complete

Floppy Disk:

Turbo Mode (45.2 KBytes/sec)	32 seconds
Secure Mode (32.2 KBytes/sec)	45 seconds

Scanner Accuracy

'VB Standard' Test-Set [1]	Turbo	359/371
	Secure	359/371
'In the Wild' Test-set [1]	Turbo	66/80
	Secure	66/80
'MtE' Test-set [1]	Turbo	1746/1926
	Secure	1746/1926

Technical Details

Product: *ViruSafe v5.4*

Serial Number: Not provided

Author: *EliaShim Microcomputers Ltd*, 5 Haganim Street, PO Box 8691, Haifa, Israel.

Telephone: +9 72 4 516111

Fax: +9 72 4 528613

Distributor (UK): *User Friendly Microsystems*, 22A Bartleet Road, Washford Industrial Estate, Redditch, Worcestershire, UK.

Price: £79 DOS only
£99 (DOS and *Windows*)

Telephone: +44 (0)527 510105

Fax: +44 (0)527 514229

Distributor (US): *EliaShim Microcomputers Inc*, 1236 West Highway 436, Altamonte Springs, FL 32714, USA.

Telephone: +1 407 682 1587

Fax: +1 407 869 1409

Hardware Used: *Compaq* 386 running at 16 MHz. The hard disk speed test was unable to be completed (see text); the floppy disk speed test measured the time to scan a 3.5-inch high-density diskette which contained forty-three files (1,446,811 bytes), all of which were executable.

[1]For complete details of the test-sets used in preparing this review, please see page 19 of January 1994 edition of *Virus Bulletin*.

PRODUCT REVIEW 2

F-PROT Professional

Dr Keith Jackson

This review discusses two versions of the same product: *F-PROT Professional* - one as supplied by *Command Software*, and the other distributed by *Data Fellows*. The scanner component of the package is developed and maintained by Fridrik Skulason, *VB*'s Technical Editor, and is from the same stable as the shareware product of the same name. The two companies mentioned above sell *F-PROT* throughout the world, though the United Kingdom is the only country where both versions are available.

Data Fellows' version of *F-PROT* was supplied on two 1.44 Mbyte, 3.5-inch floppy disks; one labelled *F-PROT Professional*, the other, *F-Scheduler*. *Command Software's F-PROT* was on a single low-density (720 Kbyte) 3.5-inch disk. Why *Data Fellows'* version is supplied on a high-density (1.44 Mbyte) floppy disk is beyond me: *Command* shows that the same software fits onto a 720 Kbyte floppy disk. This ensures that it works even on computers with no high-density disk drive. Being a *Windows* program, the scheduler is larger, and will not fit on a 720 Kbyte disk.

Documentation

The documentation supplied with each product varies enormously. *Data Fellows* provides a poorly-indexed, loose-leaf A5 manual, over 200 pages long, which explains everything in detail, and includes several *F-PROT* Update Bulletins. Explanations of use occupy 46 pages, and information on viruses known to *F-PROT* takes 163 pages.

Command supplies a small (30 page) non-indexed A5 booklet, which does little more than explain how to 'drive' the various software components. A six-page 'Addendum' is included. It is rather strange that the various command-line switches and return codes are better documented in *Command's* small booklet than in the voluminous documentation provided by *Data Fellows*.

All support details provided with *Command's F-PROT* refer to telephone numbers and contact addresses in the USA; those provided with *Data Fellows'* version, to Finland. The software author lives in Iceland, so there seems to be a long chain of distribution involved with this product.

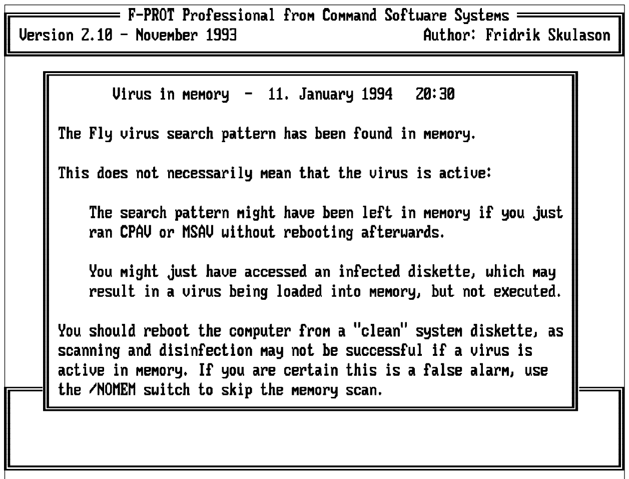
Installation

Product installation was marred by niggly problems. I first installed the version of *F-PROT* distributed by *Data Fellows*: during the initial part of the installation it stopped, refused to continue, and displayed the error message, 'The Fly virus search pattern has been found in memory … (it) might have been left … (by) *CPAV* or *MSAV*'. Neither *Central Point Anti-Virus* (*CPAV*) nor *Microsoft Anti-Virus* (*MSAV*) had been executed, so they were obviously not to blame. Detective work showed that the error was caused by executing *Sophos' Sweep* immediately before the *F-PROT* installation program. Fly is highly polymorphic, so it is unlikely that this false positive is simply a pattern left in memory.

This hitch was overcome by powering down and rebooting. *F-PROT* was then executed from floppy disk, and 'Install' selected from the main menu. Unfortunately, the only options provided by the sub-menu were Language (only English was available) and Setup. Nothing permitted installation to proceed under *F-PROT's* control and I had to resort to following the instructions for manual installation by advanced users. There is no excuse for getting installation instructions wrong: it is confusing and unnecessary.

Command's F-PROT had the missing Install option, but exhibited other foibles, such as producing an error message, 'Cannot install VIRSTOP.EXE - wrong version'. VIRSTOP is *F-PROT's* memory-resident anti-virus software program (see below). After completing automatic installation and testing the scanner, I found a small section in the manual explaining that installation should not be done this way (even though the documentation explains how to do it!), and that the file INSTALL.BAT should be used instead.

This batch file has the 'whizzo' feature of trying to execute *F-PROT*, even if the installation program has been terminated before doing anything. Not very satisfactory. It also adds and alters various files, so VIRSTOP works correctly when *Windows* is running. VIRSTOP does not now complain about a wrong version, and the VIRSTOP menu option provided within *F-PROT* works correctly if this new installation program is used. I would suggest that only one installation method be made available: the one that works.



```
F-PROT Professional from Command Software Systems
Version 2.10 - November 1993                Author: Fridrik Skulason

      Virus in memory  -  11. January 1994   20:30

   The Fly virus search pattern has been found in memory.

   This does not necessarily mean that the virus is active:

      The search pattern might have been left in memory if you just
      ran CPAV or MSAV without rebooting afterwards.

      You might just have accessed an infected diskette, which may
      result in a virus being loaded into memory, but not executed.

   You should reboot the computer from a "clean" system diskette, as
   scanning and disinfection may not be successful if a virus is
   active in memory. If you are certain this is a false alarm, use
   the /NOMEM switch to skip the memory scan.
```

The heart of the *F-PROT* package is the scanner. This is capable of detecting most common viruses in memory, reducing the risk associated with using the scanner without clean booting.

Scanning

The *F-PROT* scanner provides an initial menu showing the chosen scanning method, the disk to be scanned, the action to be taken if any viruses are found, and the files to be searched. The user can set these in any desired way before scanning commences. The scanning component of both versions of *F-PROT* appeared to be identical, and the results reported below were the same for both versions.

Three methods of scanning are provided: Secure Scan, Quick Scan and Heuristics. Secure Scan uses two different signatures for each virus, and reports the exact strain of virus found. It can disinfect some (most?) virus-infected files. Quick Scan is faster, but not as thorough. It uses only one virus signature, does not offer disinfection, and identifies a virus as being from a family, rather than detecting the exact variant. Heuristics (science-speak for guessing) follows its own rules to try to identify suspicious files.

I tested *F-PROT's* scanning speed while it searched the contents of my test PC's hard disk. Using Secure Scan, this took 1 minute 48 seconds, and just 28 seconds when Quick Scan was used - very quick indeed. For comparison purposes, *Dr Solomon's Anti-Virus Toolkit* scanned the same hard disk in 1 minute 20 seconds, and *Sophos' Sweep* in 2 minutes 16 seconds on a quick scan.

The set of viruses used for testing purposes has to be updated from time to time, and this month the test-set has been expanded by the addition of 12 parasitic viruses and two boot sector viruses. A complete list of the viruses used for testing purposes is included in the *Technical Details* section, but for the record, the new additions to the test-set are: 8888, Butterfly, Coffeeshop, Fish.1100, Halley, HideNowt, Invisible Man, Monkey, NukeHard, Quox, Satan Bug, Sibel Sheep, Starship and Willow.

Secure Scan correctly detected all bar two test samples. All 1024 Mutation Engine (MtE) samples were detected correctly. Quick Scan reduced overall detection to 86%, and prevented detection of MtE samples. The Heuristic method did not find the two undetected virus test samples - in fact, it reported nothing at all on my hard disk (a reassuring false positive rate of 0%). I am not certain what the Heuristics method does, as the documentation does not explain its 'rules' in detail. However, it took several seconds thinking about STACKER.COM (from the disk compression utility of the same name), before giving it a clean bill of health.

The two viruses not detected (8888 and NukeHard) are in this month's addition to the test-set, and are both comparatively new. Given *F-PROT's* excellent virus detection record, these omissions will no doubt be rectified soon.

I find no real fault with *F-PROT's* speed, or its accuracy. However, when a report of the viruses detected is viewed on screen, if the PageUp/PageDn keys are used repetitively, some reported infections disappear from view. This fault is not exhibited by the scanner report file when written to disk, and so must be a quirk of the report viewer. It does not always occur and I cannot pin it down precisely. Also, one MtE test sample was written into the report file as being detected twice - but only one file existed on disk.

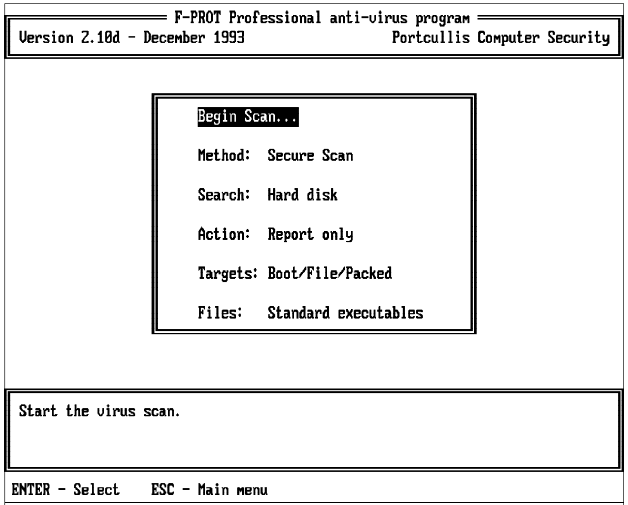
Memory-Resident Anti-Virus Software

VIRSTOP, *F-PROT's* memory-resident utility, can detect a virus-infected file not only when it is executed, but also when it is first opened (e.g. as a prelude to a file copy). It occupies only 4 Kbytes of conventional memory, creates a 19 Kbyte hidden file in the root of drive C, and has various run-time switches to configure its many features.

The impact of VIRSTOP on PC operation was estimated by copying many small files both with and without VIRSTOP present. Without VIRSTOP, a set of 41 files, occupying 1.49 Mbytes, took 27.4 seconds to copy from one hard disk subdirectory to another. The same operation took 29.4 seconds with VIRSTOP active, an overhead of 7% - far lower than most memory-resident anti-virus software programs I have tested.

The percentage of infected files detected by VIRSTOP appears to have strong links with to the virus' age (i.e. date of development/discovery) and complexity. My virus test samples are arranged in separate subdirectories, according to when they were first introduced as test samples. The detection rate of the very oldest samples was 98%; good to say the least, especially given that a recent *VB* survey of memory-resident anti-virus programs showed that such products rarely achieve a decent detection rate.

As more recently introduced test samples were inspected, the detection rate provided by VIRSTOP fell in the following sequence: 93%, 93%, 69%, 28%, 22%. This is unsurprising, as more recently developed viruses tend to be polymorphic and/or encrypted, and cannot be detected simply by scanning for known patterns. If such viruses were detected, the impact on PC operation would be much larger.



```
F-PROT Professional anti-virus program
Version 2.10d - December 1993            Portcullis Computer Security

            Begin Scan...

            Method:   Secure Scan
            Search:   Hard disk
            Action:   Report only
            Targets:  Boot/File/Packed
            Files:    Standard executables

Start the virus scan.

ENTER - Select    ESC - Main menu
```

Both versions of *F-PROT* are very similar, to the extent that screen shots appear identical except for the company name in the top right-had corner of the screen.

I did find a few quirks associated with VIRSTOP. Firstly, the file extension is used to tell whether or not a file is executable, and files are not scanned when opened unless they are thought to be executable. This is reasonable, but needs clear explanation in the documentation. Secondly, I normally use the shareware command interpreter 4DOS as a replacement for COMMAND.COM. With this in use, VIRSTOP failed to detect any viruses by inspecting files as they were opened, irrespective of the copy method used.

When the same tests were carried out using the more usual MS-DOS command interpreter (COMMAND.COM), viruses were detected when infected files were opened, but a multiple file copy always stopped after the first infected file was detected and an 'Unable to open file' error was reported. This is a feature of COMMAND.COM, not of VIRSTOP.

These actions are not the fault of VIRSTOP alone, but probably an inevitable consequence of interaction between the operating system and add-on memory-resident software. I am impressed by VIRSTOP's small size, its very low impact on normal PC operation, and its high rate of virus detection. The *Data Fellows* documentation explains that VIRSTOP uses a simple fast search algorithm, which will not detect all viruses. If such a disclaimer appears in *Command Software's* documentation, I cannot find it. Limitations of parts of the product must be brought out fully in the documentation: VIRSTOP is very good at what it does, but it is not a panacea for all ills.

Other Points

F-PROT as supplied by *Data Fellows* includes a *Windows*-based scheduling program, which permits a flexibly selected regime of scans to be imposed. It is easy to use: when first executed, it requests the pathname for the *F-PROT* executable file, and from that point on remembers the setting and any chosen times at which *F-PROT* should be executed. However, it did not remember how the Scheduler window had been re-sized, and always insisted on starting execution with the default size.

Both versions of *F-PROT* include a checksum program, albeit differently named. The *Data Fellows* checksum program is not declared as part of *F-PROT* and is only described in an easy-to-miss Appendix: this despite the fact that both checksummers look very similar.

Conclusions

Users who only know how to use mouse-driven *Windows* programs will struggle with this program. *F-PROT* does not use a mouse and is a cursor-driven DOS program which will run in a DOS box under *Windows*. This will be their loss, as *F-PROT* is one of the best scanners around at the moment. It comes packaged in various ways, and combined with various other products. It can be tailored for use interactively or through command-line switches, and most importantly is fast in execution and accurate at virus detection. These virtues are worth far more than a swish user interface.

Both versions of *F-PROT Professional* reviewed exhibit some minor problems, and the installation routine needs tidying up. The principal difference between the two versions of the products is the price, which varies by up to £100. Regardless of this, the core product is what the user is purchasing: that appears to be consistently good.

Technical Details

Product: *F-PROT*

Developer: Fridrik Skulason

Vendors:

1) *Command Software Sytems Inc.*, 1061 East Indiantown Rd, Suite 500, Jupiter, Florida 33477. USA. Tel. +1 (407) 575 3200 Fax +1 (407) 575 3026. BBS +1 (407) 575 1281.

2) *Data Fellows Ltd.*, Wavulinintie 10, FIN-00210, Helsinki, Finland. Tel. +358 (0) 692 3622 Fax +358 (0) 670156.

Availability: *MS-DOS v2.0* or higher, 512 Kbytes of RAM, 3.5- inch floppy disk drive, at least 1 Mbyte of hard disk space (optional). *F-PROT* will operate correctly on *Novell*, *3 COM* or *Banyan Vines* networks.

Version evaluated: 2.10

Serial number: None visible

Price: £59 from *Command Software*, with bi-monthly updates. £175 from *Data Fellows*, with monthly updates.

Hardware used: *Toshiba 3100SX* laptop incorporating 16 MHz 386 processor, 5 Mbytes of RAM, one 3.5-inch (1.4 Mbyte) floppy disk drive, and a 120 Mbyte hard disk,

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

The test set contains 9 boot sector viruses (Brain, Form, Italian, Michelangelo, Monkey, New Zealand 2, Quox, Spanish Telecom, VSign), and 239 samples of 150 parasitic viruses (n.b. Spanish Telecom appears in both lists). There is more than one example of many of the viruses, ranging to up to 12 different variants in the case of the Tiny virus. The parasitic viruses used for testing are listed below. Where more than one variant is available, the number of examples of each virus is shown in brackets. For a complete explanation of each virus, and the nomenclature used, please refer to the list of PC viruses published regularly in *VB*:

1049, 1260, 12 TRICKS, 1575, 1600, 2100 (2), 2144 (2), 405, 417, 492, 4K (2), 5120, 516, 600, 696, 707, 777, 800, 8888, 8 TUNES, 905, 948, AIDS, AIDS II, Alabama, Ambulance, Amoeba (2), Amstrad (2), Anthrax (2), AntiCAD (2), Anti-Pascal (5), Armagedon, Attention, Bebe, Blood, Burger (3), Butterfly, Captain Trips (2), Cascade (2), Casper, Coffeeshop, Dark Avenger, Darth Vader (3), Datalock (2), Datacrime, Datacrime II (2), December 24th, Destructor, Diamond (2), Dir, Diskjeb, Doshunter, Dot Killer, Durban, Eddie, Eddie 2, Fellowship, Fish 1100, Fish 6 (2), Flash, Flip (2), Fu Manchu (2), Halley, Hallochen, Helloween (2), Hide Nowt, Hymn (2), Icelandic (3), Internal, Invisible Man (2), Itavir, Jerusalem (2), Jocker, Jo-Jo, July 13th, Kamikaze, Kemerovo, Kennedy, Keypress (2), Lehigh, Liberty (5), LoveChild, Lozinsky, Macho (2), Maltese Amoeba, MIX1 (2), MLTI, Monxla, Murphy (2), Necropolis, Nina, Nomenklatura (2), NukeHard, Number of the Beast (5), Oropax, Parity, PcVrsDs(2), Perfume, Pitch, Piter, Polish 217, Power Pump, Pretoria, Prudents, Rat, Satan Bug (2), Shake, Sibel Sheep (2), Slow, Spanish Telecom (2), Spanz, Starship (2), Subliminal, Sunday (2), Suomi, Suriv 1.01, Suriv 2.01, SVC (2), Sverdlov (2), Svir, Sylvia, Syslock, Taiwan (2), Tequila, Terror, Tiny (12), Todor, Traceback (2), Tremor, TUQ, Turbo 488, Typo, V2P6, Vacsina (8), Vcomm (2), VFSI, Victor, Vienna (8), Violator, Virdem, Virus-101 (2), Virus-90, Voronezh (2), VP, V-1, W13 (2), Willow, WinVirus 14, Whale, Yankee (7), Zero Bug.

SUBSCRIPTION RATES

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

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

Editorial enquiries, subscription enquiries, orders and payments:

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

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

US subscriptions only:

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

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

END NOTES AND NEWS

Copies of the *Virus Bulletin* book, the *Survivor's Guide to Computer Viruses* are available from *VB*, priced at £19.95. Discounts are available for bulk purchases. Tel. +44 (0)235 555139.

For those who have always found cryptography to be a black art, do not despair - help is at hand. '**Code makers versus Code breakers: a Layman's Introduction to Cryptography**', a general interest lecture on the subject, will be given by Prof. Fred Piper, on Tuesday, 15 February, at the Lecture Theatre, Founder's Building, Royal Holloway, University of London, Egham. Tel. +44 (0)784 443004.

***Central Point* continues its buying spree**, this time acquiring *LANLord*, the workstation management product from Microcom. As a result of the deal, *Central Point* has also acquired all of the assets of *Microcom's* Client-Server Technology Group. Is no-one safe…

Cheyenne has extended the licence of *InocuLAN*, its Network-based anti-virus product, to include home and remote use. The company will make the new licence agreement available to all registered users of the product. Paul Dunford, *Cheyenne's* European Manager, commented, 'We believe that a large number of viruses were introduced by users working off-site, so really this is just to legalise unauthorised use of the software'.

SECURE Computing (the successor to *S&S International's* monthly publication *VNI*) is not now owned by the anti-virus software behemoth *S&S*, but by *West Coast Publishing*, a company owned by the magazine's editor Paul Robinson and his wife. Robinson believes that the change will be for the better, allowing him and his team to investigate a wider range of issues. Tel. +44 0(792) 324000.

The *Association of British Insurers* **has produced a Data Protection Code of Practice**, in order to outline the ways in which the personal information held on policyholders may be used. The code governs not only how personal information can be used, but also how it should be protected from both unauthorised access and destruction.

The *University of Tampere*, Finland, **plans to found a Virus Test Laboratory** in its Computer Science Department. Those wishing to have a product reviewed can pay a fee to the organisation, who will then test it. The laboratory will be run by Markko Helenius and Pertti Järvinen, and aims to provide only hard technical data on products, rather than any opinion or comment.

According to a report in *Corportate Security Digest*, **three Colorado teenagers have been arrested** for allegedly setting up a system which illegally linked people to long distance telephone lines. Detective Greg Bohlen of the *Littleton Police Department*, the teens sold the codes primarily to people linked to the computer underground. 'These kids are amazing' commented Bohlen. 'When it comes to computer technology, their knowledge and experience is amazing. But when it comes to everything else, they're out of touch.'

Two men already convicted of hacking into *Boeing Co.* and *Federal Court* computers have now pleaded **guilty to felony charges involving stolen credit card numbers and altered cellular phones**. Gig Harbor, Charles Anderson, and Costa G. Katsaniotis used a modem to access credit card records at the *Red Lion Inn* at Bellevue, then used the credit card numbers to purchase pizza and computer equipment [*The purchases are a dead giveaway. Ed.*]. Anderson and Katsaniotis face penalties of up to five years in prison, and fines of up to $250,000.

A one day seminar on the forensic examination of personal computers, aimed at internal auditors and security managers will be held at *Hambro's Conference Centre*, London, on Friday 8 April 1994. The course, led by none other than Edward Wilding and Jim Bates, costs £270+VAT. For further information, contact Rachel Forrest. Tel. +44 (0)71 344 8100. Fax +44 (0)71 344 8101.

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