March 1992 ISSN 0956-9979

VIRUS BULLETIN

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

CONTENTS

EDITOR'S DIARY

Our Man In Havana

Stepping on to the tarmac at *José Marti Airport* on the outskirts of Havana at 11:40 pm local time, *VB's* correspondent soon realised that he had made a profound and easily avoidable mistake - wearing an English wool suit in the Caribbean is plain stupid. Exhausted after a five hour delay at Madrid airport and a nine hour trans-atlantic crossing, a shower and sleep were foremost in his mind. What the hell was an Englishman doing in Cuba? 'It will be a gruelling fact-finding mission' he had told friends and acquaintances who had subsequently advised him against this trip - 'the most persuasive argument for visiting Castro's republic is the total absence of *MacDonald's* fast-food' a close friend had assured him. That, indeed, was reason enough to visit.

The bus reeked of tobacco - just about everything in Cuba smells of cigars. Havana was barely discernible. Since the loss of guaranteed Soviet oil imports, electricity and fuel have been rationed. The presidential palace, a monolithic crescent-shaped building, and a huge edifice erected as a memorial to Cuba's first national hero, José Marti, loomed large in the semi-darkness. Candles were held aloft; mourners, apparently, for an army general assassinated the day before. Then on past the imposing *University* and on to the *Havana Libre* hotel which in more decadent days had been *The Havana Hilton*.

VB's correspondent was in Cuba to speak at *Informatica '92* at the invitation of Cuba's *Comisión Nacional de Protección de Datos*, headed by the amiable and flamboyant Señor José Bidot, and *UNESCO*. At this conference, effectively a celebration of all forms of computer applications, he was to speak to an audience from across Latin-America about computers and the problems that plague them.

It became clear that Cuba, famed for its rum, sugar, and beautiful girls, intends to develop an indigenous and formidable anti-virus capability and to export that skill to other countries in South America. A dedicated team of specialists programmers, software engineers, statisticians and lawyers is preparing to turn Cuba into a world centre of excellence for virus countermeasures. In a country of only 30,000 personal computers, such effort might appear disproportionate - however, a traditional reliance on software and hardware from the former Soviet Union and virus-infested East bloc countries has exposed Cuba to this threat in a quite singular way. A devastating epidemic by the Vienna-648 virus, which swept across the island in 1988, convinced the Cuban authorities of the gravity of the threat. It was a privilege, therefore, to meet Señor Alexis Rodriguez Castedo - the first Cuban to disassemble the Vienna virus.

Questions about the origins of Cuba's hardware are gently deferred; program developers will be relieved to hear that there was no indication of any improper software use.

Senor Bidot's *Lada* pulled up outside *PAB EXPO*, the exhibition hall in which indigenous software, hardware and peripherals were on display. Battered *Chevrolets*, *Buicks*, *Dodges* and *Pontiacs* are everywhere - American chassis but with Soviet engines and transmissions...the Cuban people are very resourceful - an essential characteristic in the face of a concerted and somewhat mean-spirited embargo by the US.

Software in use by concerns as diverse as *The Ministry for Sugar Production*, *The Ministry for Tourism* and *The Ministry for Education* (*MINED*) was displayed with aplomb. Surprisingly, *The Ministry of the Interior* (*MININT*), much in evidence at the *Havana Libre*, had shied away from displaying its own particular brands of software and hardware. There are ministries for everything in Cuba - the telephone directory in the hotel listed hundreds of state organisations and ministries.

The team from *la Comisión Nacional de Protección de Datos* was celebrating. Quarantine procedures at the computer fair had reaped dividends: 214 PCs checked; 1110 diskettes checked; 19 virus contaminated hard disks; 56 contaminated diskettes. There are 19 viruses currently afflicting Cuba, the most prevalent being 1575 (Green Caterpillar), Dark Avenger, Italian, NoInt, New Zealand and Yankee. Rarer specimens include Brain and SVC 5.00. Without doubt a successful hunt which called for a totty or two, and soon the white rum, 70% proof, was circulating. Ominously, 27 of the 75 infections discovered were caused by the Michelangelo virus - if it's that prevalent in Cuba, thought the correspondent, then March 6th will be a day of reckoning worldwide.

La Comisión uses a mixture of detection software; the team are familiar with the offerings of Dr. Solomon and Mr. McAfee but have also developed a suite of 'home grown', tailor-made software aimed specifically at the viruses known to threaten Cuban computers. This decision is based on the realisation that the commercial scanners from Europe and the United States are not infallible. Indeed, a version of the Terminator virus, believed to have been first discovered in Cuba went undetected by *SCAN*, *FINDVIRUS* and *SWEEP*. The principal components of the Cuban anti-virus armoury are: *DETEC* (a virus-specific scanner); *LISTARC* (a CRC generic checker); *CVP* (a memory-resident virus-specific sentinel); and *CVS* (a detection and recovery program). In addition an excellent menu-driven virus encyclopaedia in Spanish is available which contains a wealth of information about viruses and anti-virus tools and techniques. All of the software is made available free of charge on a national basis.

PAB EXPO was to be the site of a relatively historic event - the first-ever computer virus workshop in Latin America took place there on 20th February. The delegates were instructed in diagnostic techniques and two live viruses, Dark Avenger and Brain, were demonstrated by the ever-watchful and expert instructors, Señors Carvajal and Olivera. A meeting at the *Palace of the Young Peoples' Computing* followed (there are lots of 'palaces' in Havana). Opened by Dr. Fidel Castro in 1991, these clubs are designed to make computing accessible

to the people. Ethics and law were the principal topics of conversation. The *Young Clubs (Joven Club*) extend the length and breadth of Cuba and computers (of which there are 1,000) are made available to all - even in the remote mountainous regions in the West of the island there are 'mobile computers'. With such accessibility must go responsibilty - part of the work of *la Comisión Nacional de Protección de Datos* is to ensure that young programmers do not stray into bad habits.

A leisurely drive to the Varadero beach, site of Cuba's burgeoning tourist industry to which thousands of Canadians and Germans flock each year, marked the end of an extraordinary week. Columbus' quincentenary, the house in which Ernest Hemingway penned *The Old Man and the Sea*, a genuine Scud missile (now a harmless monument but the cause of some concern to President Kennedy in 1961), the boat in which Castro and his eighty men sailed from exile in Mexico, the various titillations of the famous *Club Tropicana* described with such precision by Graham Greene - all sights and sounds veiled to most Englishmen. Swimming in the warm Caribbean water with the prospect of all that duty-free rum and *Monte-Cristo* cigars, *VB's* correspondent reflected that this particular fact-finding mission had been rather less gruelling than anticipated.

Viruses Detected in Cuba

The following viruses are currently prevalent in Cuba:

Vienna 658, Italian, Jerusalem, Cascade, Yankee Doodle, Dark Avenger, Michelangelo, New Zealand, Brain, Vacsina, Disk Killer, Amstrad 847, W13 534, Vienna 643, Terminator, 1575, Flip (Omicron), NoInt (Bloomington), SVC 5.00

PAB EXPO 'Virus Hunt'

Statistics accrued following a 'virus hunt' at the PAB EXPO computer fair, Havana, 17-22 February.

Hard disks inspected	214
Diskettes inspected	1110
Hard disks contaminated	19
Diskettes contaminated	56

Virus Types and Prevalence

Michelangelo	27
1575	24
Italian	9
NoInt	5
Dark Avenger	3
SVC 5.00	3
New Zealand	2
Terminator	1
Yankee	1

Courtesy of *la Comisión Nacional de Protección de Datos*

HEADLINES

The Onward March of Michelangelo

The heavens have aligned in the firmaments to make March 1992 a nerve-wracking time for PC Support people.

Extensive media reports about the Michelangelo virus (it triggers on March 6th and trashes both hard disks and diskettes) have stimulated an international detection frenzy which should help limit the accumulative amount of destruction the virus wreaks.

The virus appears to be widespread - the alarm bells started ringing when the news broke that *Leading Edge Products* of Westboro, Massachussets had shipped 500 PCs, the fixed disks of which were contaminated. The PCs had been distributed from the company's warehouse between the 10th and 27th December of last year. Company spokeswoman Susan Zephir said that the contamination was 'inadvertent' and that sabotage was not suspected.

On February 1st 1992 *DaVinci Systems* of Raleigh, North Carolina announced that a recent shipment of the company's *eMAIL 2.0* demonstration disk was contaminated with the virus. Approximately 900 customers and resellers had received the disks. By this time the conspiracy theorists were awakening; two major incidents of mass virus distribution was unusual and the DaVinci/ Michelangelo renaissance connection appeared to be more than a coincidence, to many contributors to the *Virus-L* conference. Indeed, *Virus-L* has veritably hummed with talk of Michelangelo, its distribution and effects. Reports of the virus being widespread on university campuses, a further report of a suspected shrink-wrapped infection (*Meridian Data* CD-ROM extension software) and the announcement by Vesselin Bontchev that the *University of Hamburg* had received 28 mailbags containing requests for the *Virus Test Center's* free Michelangelo detection and recovery program (as announced on German TV and radio by Professor Klaus Brunnstein) fuelled the prevailing sense of fear and loathing.

The virus has been detected throughout Europe (notably in the UK, Croatia and Slovenia), the United States, Canada, Australia, New Zealand, the Middle East, the Far East and, as reported in this month's editorial, has even found its way to the Caribbean.

Of technical interest, there is a bug in the virus which means that it does not trigger on most XT machines. The virus intercepts BIOS interrupt 1AH in order to ascertain the system time as part of its trigger routine. This interrupt is used only by AT-class machines and above and is not suported by XTs with a single exception, the relatively rare Turbo-XT. The virus failed to trigger on a somewhat delapidated test XT. However, this will be of small comfort to most businesses which invariably use ATs and PS2s.

An interesting feature of Michelangelo is that it only infects diskettes in drive A:, not B:. In effect, there are two isolated populations - one on machines with a 5.25 inch disk drive as A: and the other on machines with a 3.5 inch drive as A:.

One ludicrous piece of advice, reportedly published in a PC magazine which *VB* has not been able to trace, instructed users to forward their system clocks to March 6th to see whether the machine was infected! The virus was detected on at least three PCs using this method - it triggered on all of them causing irreparable damage! Legitimate stop-gap remedies include not using the PC at all on the trigger date or booting the PC from a clean, write-protected system disk (and nothing else) on March 6th. Logically, forwarding the system clock to March 7th should work, but due to the differences between real-time clock implementations, this method cannot be guaranteed to avert the trigger routine. Early and reliable detection is the prescribed method for avoiding the virus. Stop-gap methods do not disinfect the machine or diskettes, which leaves the virus to propagate through diskette interchange - the virus can thus spread and trigger again on March 6th 1993.

Free remedial software was made available during late February on *Compuserve* including dedicated Michelangelo scan and disinfection programs from *Symantec*, *Central Point* and *Trend Micro Devices*. *Total Control*'s CUREMICH.COM, was made available on various bulletin boards in early February. A special *Compuserve* forum 'GO MICHELANGELO' facilitated program download.

March 13th 1992 also happens to be a Friday and isolated reports of the Jerusalem virus triggering are expected. The virus is relatively harmless (in comparison with Michelangelo) and readily detectable using any of the commercial or shareware scanners that are available. If a Jerusalem infected program is executed on the trigger date, the virus will delete the program. Disinfection of the virus is best achieved by identifying infected files with a scanner, deleting them using the DOS delete command and restoring from write-protected master software. No incidents of the Jerusalem virus triggering were reported to *VB* on either Friday 13th December 1991 or Friday 13th September 1991.

Finally, the Maltese Amoeba virus is set to trigger on March 15th - the Ides of March. March 15th 1992 is a Sunday, which will have the effect of reducing the number of incidents reported. Moreover, since announcing its unwelcome presence on November 2nd 1991, the virus has lapsed into relative obscurity. There have been no subsequent reports of it circulating in the United Kingdom although Maltese computer users are advised to remain extremely vigilant (see page 11).

Further Information

Michelangelo: *VB*, January 1992, pp. 13-14

Jerusalem: *VB*, July 1989, pp. 10-11; August 1989, p. 10; October 1990, p. 8; May 1991, p. 3

Maltese Amoeba: *VB*, December 1991, pp. 13-16

LETTERS

Dear Editor,

I was appalled to read of Dr. Fred Cohen's intention to organise a virus writing contest (*VB*, January 1992, pp. 4-6) and I felt that his reply to your item should not pass unremarked.

While most of us have become more and more concerned with the practicality of protecting users against viruses, Cohen's theoretical pronouncements have gradually been reduced to the status of a mildly interesting sideshow. He blithely suggests that 'some benevolent viruses have been in safe use for 4 years with no ill effects'. Since his definition of a virus includes even simple disk copying programs, I would ask for his understanding of such terms as 'benevolent' and 'ill effects', perhaps he might even tell us of the particular 'viruses' he has in mind? I am sure that if Dr. Cohen had discovered a genuinely beneficial use for virus code, the world would have been informed.

It seems that total irresponsibility is an essential character trait for those who create and publish virus code. It also seems that such irresponsibility is not easily eradicated. Dr. Cohen's reputation was built on his early experiments with the virus concept and this was accepted at the time as a justifiable research effort.

However, using his own definition of viral activity, Cohen has now introduced a new viral strain - in publicising his intention to hold this ludicrous competition, he becomes an active stimulus in helping virus code to proliferate. He has once again established a precedent - a potential source of computer viruses which does not actually need a computer to get started. This time, though he attempts it, the motive cannot be justified as research.

The specious arguments that he presents to support his ideas ignore the fact that computing technology is now practically universal. His approach is analogous to medical researchers seeking a cure for AIDS by holding a competition to design an anti-HIV virus, knowing that millions of people had sufficient technology to enter!

No, Dr. Cohen, there are beneficial uses for dynamite, but that is no excuse for holding a bomb-making competition, particularly when so many people already have all the ingredients.

Finally, I would like to make it clear that there is absolutely no connection between *ASP(UK) Ltd.* and Dr. Cohen's company.

Jim Bates,
Chairman
ASP(UK) Ltd.

Dear Editor,

I read with interest your comparative review ('Scanners - The Acid Test') in the January *VB*. The IBM scanner missed five of the 34 viruses; it is some consolation to us that you happened to catch *VIRSCSAN* at the end of a product-release cycle. The new version, 2.1.9, which was coming out as you went to press, detects all 34 of the viruses you tested against (assuming that we are using the same names for the same viruses). 2.1.5 was, of course, released before the Maltese Amoeba appeared; the other four viruses are ones that we have never seen in the wild ourselves, but they've been added as part of regular updating (not just because of your review!).

I was curious about some of the viruses that were included, and not included, in your test set. The Old Yankee viruses, for instance, have been around for some time, but I have never heard, even at second or third hand, a report of a real infection. The Yankee Doodle viruses (a family that is unrelated to the Old Yankee viruses), on the other hand, are common (at least the 2885-byte variant), but were not included. (I also don't believe that the Whale is actually spreading in Australia; it works so badly! My guess is that a schoolboy or two has a copy, and sneaks it onto a machine now and then.)

In general, I would like to publicly applaud the trend towards testing scanners against viruses that are actually active in the wild. This should be a much better measure of the usefulness of the products being tested. Perhaps an even more accurate test might involve weighting the results according to how common each of the viruses involved are. Of course, then everyone detects the Stoned, 1813, Joshi and Cascade! Perhaps ease of use might turn out to be at least as important as detecting the Spanz virus!

Dave Chess
IBM T. J. Watson Research Center
New York

Dear Mr Wilding,

As editor of the *Virus Bulletin*, your job of publishing reviews that are considered fair, unbiased, and accurate by your entire readership, is challenging at best. Increasingly, those concerned with the growing threat of virus infection look to *Virus Bulletin* as a reliable source of objective information and insight. From a vendor's perspective, it is, therefore, both frustrating and disappointing when we feel our product has been represented negatively to what we know is a trusting and loyal readership. Such is the case with your recent review of *Untouchable*.

Despite our differences, I would like to offer a sincere thank you for allowing us to see and respond to the review prior to its publication. Likewise, incorporating a few of our remarks helped temper some of the inaccuracy of the initial review.

It has long been my stance that editors should work with publishers in order to ensure 100 percent accuracy of information presented to its readers. Your willingness to work with *Fifth Generation Systems* on the Untouchable review (and I am sure with other vendors in their product reviews) is certainly a policy other publications should implement.

As discussed, we will send an updated version of the virus-specific scanner, *UTSCAN*, for reevaluation and publication in the March issue of *Virus Bulletin*. We hope you will work with us, once again, to guarantee that the complete functionality of the product is thoroughly examined and accurately presented.

In closing, thank you for your interest in *Untouchable* and for your responsiveness to our concerns. We look forward to a strong and mutually beneficial working relationship in the future.

Sincerely,

Barry L. Bellue, Sr.
President and CEO
Fifth Generation Systems
Baton Rouge, Louisiana, USA

Dear Mr. Bellue,

Thank you for your letter and obvious appreciation of the sensitive task faced by any editor when dealing with powerful vested interests. You imply that inaccuracy exists in the review of Untouchable *as published - it would be helpful if you could be more specific as to what, exactly, remains incorrect. Since numerous telephone conversations between* VB *and* FGS *and pages of fax communication were expended on ironing out inaccuracies (which were legion in the copy prior to publication), I am somewhat surprised that anything is still amiss. I would contend that the review published last month contained no factual inaccuracy - subsequent independent checks upon the reviewer's reported results confirmed their correctness.*

I am sure that the next release of UTSCAN *will surpass expectations and, likewise, look forward to a cordial relationship with all at* FGS. *Ed.*

Dear Mr. Wilding,

Thank you for your letter dated 7th January concerning *PC Fun* and comments regarding the virus which is contained within the cover-mounted disk. [*VB*, February 1992, pp. 6-7]

We were made aware of the virus and that it was relatively harmless by the magazine's distributor *COMAG* on the 7th January. However, we did decide that in the interests of our customers that all copies should be removed from sale and returned to the publisher immediately.

Whilst this action is after the event it does penalise the publisher financially and in turn should improve their security controls at the duplication stage.

My thanks for drawing this matter to our attention.

R. J. Francis
Retail Merchandise Director
John Menzies Retail Division
Edinburgh, Scotland

Dear Sir,

I read with interest Dr. Keith Jackson's review of the *ViruGuard* product in the February 1992 edition of *Virus Bulletin*. While agreeing with its conclusions and his review in general, I must take issue with one assertion he makes.

He states on page 26 that 'Being able to choose a password that is already in use is weak even when it is associated with a particular ID.' How does he propose that a security system should advise a user that the chosen word is unacceptable without implicitly compromising someone else's password?

While it may be argued that such a compromise is acceptable on a system with thousands, or perhaps even only hundreds, of users, this is only so where User IDs are not sequential or easy to guess. The only reason for not allowing users to have identical passwords is where the (hopefully) encrypted list of passwords can be accessed by users. It might then be necessary to prevent duplicates in order to thwart a dictionary attack. However, proper controls on the number of invalid logins permitted should prevent the need for even this.

In any case, these numbers of users are not representative of PCs. A typical PC security system deals with small numbers of users who know each other and, most likely, each other's User IDs. Compromising another user's password, as Dr. Jackson suggests, represents an open invitation to hack. No reputable security system that I am aware of imposes such a restriction.

Yours faithfully,

David Henretty
Principal Engineer (Security Group)
Apricot Computers Ltd.

The editor welcomes your letters, which can be sent by post or to fax number +44 (0)235 559935

IBM PC VIRUSES (UPDATE)

Updates and amendments to the *Virus Bulletin Table of Known IBM PC Viruses* as of 1st March 1992. Hexadecimal patterns may be used to detect the presence of the virus with a disk utility or preferably a dedicated scanner.

Type Codes

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

Seen Viruses

1355 (temporary name) - CR?: 1355 bytes, awaiting analysis.

```
1355            8B04 8ED8 BE00 00B0 2EB4 803A 0475 1BB0 3A3A 4401 7514 B026
```

Ada - CR: A 2600 byte virus, reported to have originated in Argentina, which targets the *PC-CILLIN* anti-virus package.

```
Ada             4802 0074 0F80 FC41 741B 80FC 1374 163D 004B 7406 9D2E FF2E
```

Alabama 2 - ER: Slightly modified version of the original virus, but detected by *VB's* Alabama pattern.

Ambulance-B - CN: 796 bytes long, similar to the original but with a few insignificant modifications.

```
Ambulance-B     0001 8A07 8805 8B47 0189 4501 FFE7 CBE8 DE00 8A84 2804 0AC0
```

Black Jec (formerly Bljec) - CN: Four new variants of this virus have been found, but they are all detected by the pattern published previously. The differences seem to be caused by the fact that a different assembler has been used to assemble the source code.

Cascade 1701-F - CR: Very closely related to the 1701-A variant, but the encryption routine has been changed.

```
Cascade 1701-F  012E F687 2A01 0174 0F8D B74D 01BA 8206 3134 3114 464A 75F8
```

Cascade 1706 - CR: This variant seems to be based on the 1704 byte variant, but it has been changed slightly and reassembled.

```
Cascade 1706    3001 F687 2901 0174 0F8D B74B 01BC 8806 3134 3124 464C 75F8
```

CAZ - CER: 1204 bytes. Awaiting analysis.

```
CAZ             8BEC 7207 8366 0AFE EB08 9083 4E0A 01EB 2390 50B4 2F9C 2EFF
```

Criminal - CN: An encrypted message in bad English is found in this Dutch virus, which urges the user to turn himself in for illegal copying. Partially analysed. Suspected of being destructive.

```
Criminal        01EE C604 E989 4401 C744 03FF 20B4 42B0 008B 9E1F 0BB9 0000
```

Danish Tiny-251 - CN: This virus seems to be derived from the 163 byte variant, but is not particularly interesting.

```
Danish-251      8BFA B903 00CD 2180 3DE9 7407 B44F EBDC E988 00B8 0057 CD21
```

Danish Tiny-Stigmata - CN: A 1000 byte version, with a considerable part of the virus' body taken up by a greeting to various virus writers and anti-virus developers.

```
Danish-Stigmata 5053 5156 8B9C EB04 81C6 5C01 B98D 0390 D1E9 7301 4E8B FEAD
```

DM-330 - CR: This encrypted virus contains text stating it is version 1.05 of the DM virus, but it is considerably different from the earlier versions. Only a partial search string (which includes wildcards) is possible.

```
DM-330          B8?? ??B9 3701 BE?? ??50 8034 ??46 E2FA C3
```

Even Beeper - ENP: This virus is highly unusual. It is a companion virus, which creates a COM file for every EXE file it 'infects'. The COM files are structurally EXE files, written in a high-level-language, but the length is variable, and they have been compressed with the *LZEXE* program. As a result it is impractical to use a signature to detect infected files.

EUPM, Apilapil - CER: An encrypted, 1731 byte virus. Awaiting analysis.

```
EUPM            2E8C 0601 008C C88E D8B9 A006 BF03 002E A000 002E 0005 47E2
```

Fichv 2.0 - CN: Very similar to the more common 903 byte variant, but only 896 bytes long.

```
Fichv 2.0       B801 35CD 218C 0629 0189 1E2B 01B8 0335 CD21 8C06 2D01 891E
```

Frogs B - CN: A very minor variant of the earlier Frogs (Frog's Alley) virus, and detected by the same pattern.

Gotcha-D - CER: The smallest member of the Gotcha family, 627 bytes long.

```
Gotcha-D        9C3D DADA 742E 5251 5350 5657 1E06 80FC 3E75 04B4 45EB 073D
```

Grune - CR: The name of this virus is derived from an encrypted text message inside it, which refers to the Green party of Switzerland. Infected programs grow by 1241 bytes.

```
Grune           3601 0026 C606 0000 4D5E 5681 C6D5 0483 C360 5307 8BFE FDB9
```

Hafenstrasse-1689 - EN: This 1689 byte, updated version of the Hafenstrasse virus differs considerably from the original. It contains a copy of the Ambulance virus, which it will 'drop', infecting COM files with that virus, but the Hafenstrasse virus itself only infects EXE files. Detected by the pattern for the 809 byte variant.

Halloween - CEN: The virus triggers on Halloween (October 31st) and truncates files to 666 bytes.

```
Halloween       6F77 6565 6E55 89E5 B8B8 009A 4402 5701 81EC B800 8D7E FE16
```

Itti-191 - CN: A primitive overwriting virus, which displays the text 'EXEC failure' when it has infected a program. The virus will not attempt infection if *Flushot+* is active in memory. A related 99 byte virus does not check for the presence of *Flushot+*.

```
Itti-99         998B CAB8 0042 CD21 B440 B963 00BA 0001 CD21 B43E CD21 9DC3
Itti-191        7415 B44E B927 00BA 8C01 CD21 7215 E81D 0075 04B4 4FEB F3B4
```

Jabberwocky-615 - CR: Detected by the Jabberwocky pattern.

Jerusalem-1244 - CER: One of the shortest Jerusalem variants, only 1244 bytes long.

```
Jeru-1244       2638 05E0 F906 0E07 1F8B D7B8 004B 83C2 03BB 3F00 9C26 FF1E
```

Jerusalem-1735 - CER: A 1730/1735 byte variant, which appears to be related to the 1767 variant. Partially analysed.

```
Jeru-1735       2638 05E0 F98B D783 C203 B800 4B06 1F0E 07BB 3500 1E06 5053
```

Jerusalem-Barcelona - CR?: Unlike most other members of the Jerusalem family, this 1792 byte virus does not seem to infect EXE files. It is of Spanish origin, and seems to be politically motivated.

```
Barcelona       2638 05E0 F98B D783 C203 B800 4B06 1F0E 07BB 2C00 5351 5256
```

Jerusalem-Moctezuma - CER: A 2228 byte polymorphic (variably encrypting) variant of the Jerusalem virus, which contains the text 'Moctezuma's Revenge'. Only a short search pattern is possible.

```
Moctezuma       062E 8F06 0201 1E2E 8F06 0001 0E07 0E1F BF
```

Jerusalem-Nov 30. - CER: This 2000 byte variant activates on November 30th, instead of Friday the 13th.

```
Jeru-Nov 30.    2638 05E0 F98B D783 C203 061F 0E07 BB30 00B8 004B 2EFF 1E1C
```

Jerusalem-Sub Zero, **Skism11** and **Skism12** - CER: Three 1808/1813 byte variants, which are detected by the Captain Trips pattern.

KO-408 - CR: 408 bytes. Awaiting analysis.

```
KO-408          5B53 B802 4233 C9BA FFFF CD21 8BD0 33C9 B800 42CD 210E 1FB4
```

Leprosy-C2 - CEN: A primitive 666 byte overwriting virus. When run, it displays the message 'Program too big to fit in memory'. This virus is available on virus BBSs under the name of 'Durango', but in fact it is just a minor variant of the Leprosy-C virus.

```
Leprosy-C2      53E8 1000 5B90 B99A 02BA 0001 B440 CD21 E801 00C3 BB34 018A
```

Leprosy-Viper - CEN: This 840 byte variant is similar to the Plague variant, but it uses a slightly modified encryption algorithm. Just like the C2 variant it is only found on virus BBSs, and is not a serious threat.

```
Leprosy-Viper   BB3A 018A 2732 2606 0188 2790 9090 4381 FB82 047E EEC3
```

Manuel - CR: This 957 byte virus contains the text: 'Soy un Manuel Virus de tipo C'. Partially analysed.

```
Manuel          F9C3 A675 FBF8 C3FC 268A 25AC 3C00 7415 3AC4 75F7 5747 56E8
```

Marauder - CN: This virus contains text which indicates it is written by the authors of the Phalcon and Skism viruses. It is polymorphic (i.e. it employs self-modifying encryption) and no simple search string is possible from the variable decryption routine. The virus is 860 bytes long.

Multiface, Portugese - CR: This is a 1441 byte virus from Portugal. It is reported to display multiple 'smileys' on the screen.

```
Multiface       8ED8 58C6 075A C747 0100 0089 4703 5B8D B700 00BF 0000 0E1F
```

Murphy-Tormentor - CER?: This virus infected only EXE files in testing, but it seems to contain code to infect COM files too. Detected by the HIV pattern.

Padded - CN: A 1589 byte virus which is padded with a large block of zero bytes which serve no apparent purpose.

```
Padded          BA00 00CD 215A 4AB4 40B9 0300 CD21 B802 42B9 0000 BA00 00CD
```

Pathhunt - EN: Even though this virus only infects EXE files they are infected as if they were COM files - the first few bytes overwritten with a jump to the virus body. Partially analysed.

```
Pathhunt        03FD 8A0D 2ED2 0F59 43E2 EEEB 1DBB 1A01 E866 FF03 DDB9 F603
```

Phalcon-Ministry - CN: Encrypted, 1168 byte variant of the Phalcon virus.

```
Ministry        BE15 0103 3606 018A 24B9 5504 81C6 2E00 8BFE AC32 C4AA E2FA
```

SBC - CER: A polymorphic 1024 byte virus, with full stealth capability - hiding file size increases as well as file changes when active. This virus is not just a laboratory virus - it is spreading in Canada and the US. No search pattern is possible.

Screamer - CER: A 711 byte virus, which contains the text 'Screaming Fist'. Partially analysed.

```
Screamer        89D7 B02E B9FF 00F2 AEE3 2889 FE26 AD25 DFDF 3D43 4F74 113D
```

Seventh Son - CN: Two slightly modified versions of the 332 byte virus reported earlier, 350 and 284 bytes long.

```
Seventh Son350  73F3 1F5A B824 25CD 215A B801 33CD 210E 0E1F 07B8 0001 50C3
Seventh Son284  56A5 A55E B800 33CD 2152 9940 50CD 21B8 2435 CD21 5306 8D94
```

Suriv 1-Argentina - CR: This variant of the April 1st virus was reported in Argentina. It is 1249 bytes long and may display messages on various dates which are of patriotic significance in Argentina.

```
Argentina       0E1F B42A CD21 81FA 1905 7415 81FA 1406 7415 81FA 0907 7415
```

Tic - CN: A simple, 109 byte virus, which does nothing but replicate.

```
Tic             B44E EB06 B43E CD21 B44F 0E1F CD21 B91E FE72 288B D1B8 023D
```

Traceback-3029 - CER: This is the first new member of the Traceback family to appear in a long time. Not yet fully analysed, the virus does not appear to be significantly different from the other known variants.

```
Traceback-3029  B419 CD21 89B4 5101 8184 5101 5F08 8C8C 5301 8884 E300 E8E3
```

Trivial-38 - CN: Yet another 'minimalist' virus which does nothing but replicate by overwriting the beginning of other programs.

```
Trivial-38      3DCD 2193 B126 BA00 01B4 40CD 21B4 4FEB E6C3 2A2E 636F 6D00
```

Vienna-618 - CN: Detected with the Vienna (1) pattern.

Vienna-621 - CN: This variant is detected with the Vienna (4) pattern. It is similar to the original virus, but instead of overwriting programs with an instruction that resets the computer, it overwrites them with the instruction JMP C800:0000, which may cause a low-level format of the hard disk on certain machines.

Vienna-Dr. Q 1028 - CN: Similar to the 1161 byte virus described last December and detected by the same pattern. 1028 bytes long.

Vienna-Infinity - CN: A 732 byte Vienna variant with an unusual feature - it will not infect files if the PSQR virus is in memory.

```
Infinity        ACB9 0080 F2AE B904 00AC AE75 EDE2 FA5E 0789 7C59 908B FE83
```

Vienna-Mob 1a - CN: A 1024 byte Canadian member of the Vienna family.

```
Vienna-Mob 1a   ACB9 0080 F2AE B904 00AC AE75 EDE2 FA5E 0789 BCCB 008B FE81
```

Vienna-Parasite - CN: Yet another Vienna variant of Canadian origin - 1132 bytes long. Version 2B is only 903 bytes long.

```
Parasite        ACB9 0080 F2AE B904 00AC AE75 EDE2 FA5E 0789 BCF9 008B FE81
Parasite 2B     ACB9 0080 F2AE B904 00AC AE75 EDE2 FA5E 0789 7C42 908B FE83
```

Vienna-Viperize - CN: Another unremarkable Vienna variant, 934 bytes long.

```
Vienna-Viperize FC8B F290 83C6 0A90 90BF 0001 90B9 0300 F3A4 908B F2B4 30CD
```

Violetta - CR: An abominably written 3840 byte virus which is outstandingly inept. Partially analysed.

```
Violetta        B425 B0FF 061F 89DA CD21 0E1F B425 B021 BA00 03CD FFB8 F125
```

Void Poem - CR: A strange virus, with a considerable portion of its 1825 byte body containing an encrypted poem. Awaiting analysis.

```
Void Poem       0AE0 B9CB 0430 2547 E2FB BAD5 04B8 0125 CD21 0402 CD21 C3
```

We're here - CN: An 836 byte virus. Awaiting analysis.

```
We're here      B905 00CD 21BF 8600 B090 B90F 00FC F3AA B442 33C9 33D2 8B1E
```

Wonder - EN: An overwriting virus, 7424 bytes long, which appears to be written in *Borland C++*.

```
Wonder          83C4 0856 B800 1D50 B801 0050 FF76 04E8 2F06 83C4 0856 E8E5
```

XPEH - CER: Probably related to the Yankee virus, as it is detected by the Yankee pattern, but modified considerably - 4016 bytes long. Of East-European origin.

Yafo - CN: A 328 byte virus, which contains the text 'Maccabi Yafo Alufa !!!'

```
Yafo            03F5 BF80 00B9 8000 FCF3 A4C3 B802 3DCD 2172 538B D8B9 0300
```

VIRUS PREVALENCE TABLE - JANUARY 1991-JANUARY 1992

Confirmed virus incidents reported to *Virus Bulletin* between January 1991 and January 1992. Note that these figures indicate incidents (i.e verified reports) and not the number of infected PCs involved in each case. The viruses are listed in order of prevalence with the total number of incidents caused by each specimen recorded and a percentage of the total reports allocated. Monthly totals and an accumulative total over the thirteen month period are shown. The table is not completely accurate as many virus incidents reported to *VB* during this period went unlogged. These figures include virus reports from countries outside the United Kingdom - regional statistics started to be collated in October 1991. Statistics from other agencies involved in virus control are not shown. Percentages are approximate.

VIRUS	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC	JAN	TOTAL	%
New Zealand II	3	9	4	4	9	5	7	10	6	8	8	5	8	86	24.93
Form				1	1	1	4		2	9	8	7	11	44	12.75
Tequila					1	3	1	5	1	4	3	2	10	30	8.70
Cascade		5	1		2	2	2		4	2	2		2	22	6.38
Joshi			1	2	2				2	5	2	1	3	18	5.22
Eddie II					3	7	1	3			1	1		16	4.64
Jerusalem	1	1	2		1	1	3	1	2	1			2	15	4.35
Spanish Telecom					1	2	1	1		4	2		3	14	4.06
4K			1		3	2	2	1	1	2				12	3.48
Vacsina		1	1		3	2	1						2	10	2.90
Yankee	1	2				3	1		1			1		9	2.61
Michelangelo									2	3	3		1	9	2.61
Nomenklatura						1	1	2	2	1	1			8	2.32
1575							4	1				1		6	1.74
Italian			1		1	1			1					4	1.16
Flip							1			1		2		4	1.16
Keypress			1				1					1	1	4	1.16
Dark Avenger		1				1					1			3	0.87
Syslock		1	1			1								3	0.87
Murphy														2	0.58
Slow		1	1											2	0.58
Plastique					1		1							2	0.58
Aircop						2								2	0.58
SPANZ							2							2	0.58
Maltese Amoeba											1	1		2	0.58
AntiCAD 4							1						1	2	0.58
Halloechen	1	1												2	0.58
Printscreen	1													1	0.29
VIRDEM		1												1	0.29
777			1											1	0.29
Liberty			1											1	0.29
PcVrsDs			1											1	0.29
Music Bug				1										1	0.29
Disk Killer				1										1	0.29
Vienna 2					1									1	0.29
Yale					1									1	0.29
Azusa						1								1	0.29
Do Nothing							1							1	0.29
Kitty							1							1	0.29
TOTAL PER MONTH	6	22	15	8	31	36	36	25	24	40	32	22	49	345	100

FIELD REPORT

John de Giorgio

Maltese Software Under Siege

Malta tends to be regarded as the island of sunshine, history and the *Knights of Malta*. However, developments in the computer virus scene are once again projecting the island onto the world stage. It is probably true to say that the Maltese computer-using community has suffered more extensively from virus-inflicted damage than any other nation worldwide.

As in most countries, computer viruses have proliferated in Malta. Minimal protection against software piracy and a close-knit community have also contributed to a level of virus infection disproportionate to the size of the island.

In addition to the common imported viruses, Malta has developed its own 'home grown' viruses, the most famous of which are the *Casino de Malte* (*VB*, March 1991, pp. 15-16) and the *Maltese Amoeba* virus (*VB*, December 1991, pp. 15-16). When these first triggered in 1991, the effects were felt well beyond these shores.

In such a small community (population 350,000) where most people active in the computer world know each other, it has proved surprisingly difficult to identify the author(s) of these viruses. As already reported in *Virus Bulletin* (December 1991), the Maltese Amoeba virus appears to have originated at the *University of Malta*, unless this is a well planned deception. Indications are that the Casino de Malte was written by a different author.

The problem has become so serious that detection software has had to be developed locally in Malta. The most widely used of these is a scanner called *SCANAM* written to detect the Maltese Amoeba virus, which it does with varying degrees of success. The major international scanners have also been updated to detect these local viruses.

People active in the virus field range from companies distributing anti-virus software to individuals who burn the midnight oil as a hobby in order to reverse-engineer viruses on their home PCs.

One of the most active members of the anti-virus community is Arnold Stellini, the author of *SCANAM*. Stellini runs a local bulletin board and his interest is probably motivated more by self-preservation than anything else. *SCANAM* was made available free of charge for download within days of the Maltese Amoeba triggering.

Many people have pointed the finger at Stellini as a suspected author of the Amoeba citing the speed with which his scanner appeared as suspicious. The latest version of his scanner has failed to detect Amoeba whereas the commercial *SWEEP* scanner succeeded - this has been regarded by some people as adding yet more fuel to the fire. But the suspicions are not supported by any logic - why did he not try and make money out of *SCANAM*?

The chief protaganist of the home virus 'crackers' is David Cefai, who by day is the chief brewer at the local brewery. At times of crisis, he has burned his fair share of midnight oil revelling in the process of disassembly and virus analysis. Cefai is at his happiest when immersed in assembly code - the more complex the better.

At present, the virus threat from within appears to be greater than the threat from overseas. Most virus infections are caused by the Casino de Malte and the Maltese Amoeba. Cases of Jerusalem, Italian, New Zealand and Cascade are fairly common, but, for instance, I have not heard of any reported instances of Michelangelo.

Devastation and Paranoia

On November 1st 1991, Maltese Amoeba wreaked havoc in Malta. The virus had been circulating undetected for seven months. Prior to triggering, end-users had felt secure using the various international commercial and shareware scanners - attitudes have since changed! The realisation that we have at least one deviant programmer in our midst has introduced a degree of paranoia amongst Maltese computer users. We continue to use the panoply of US and European detection software but at the same time realise that local Maltese programming and disassembly skill is a first line of defence against our resident virus writer(s).

The Maltese Amoeba caused destruction at local insurance companies, shipping lines, government departments, manufacturing plants and at least one bank.

The most recent trigger date of the Casino de Malte virus (January 15th 1992) also resulted in significant data loss. A large number of the attacks resulted from infected *Novell NetWare* boot disks distributed to clients by a local supplier.

Trigger dates are ringed in red ink on most PC user's calendars (Maltese Amoeba: 15th March and 1st November and Casino de Malte: 15th January, 15th April and 15th August). When these trigger dates approach, clients start ringing for support and system dates are altered. Perhaps the best story followed from a circular my company, *Shireburn Ltd.*, distributed to clients warning of the impending trigger date of January 15th. The day after the trigger date, I was talking on the telephone to the IT supervisor for a large local organisation. He reported no unusual activity on the trigger date. I enquired as to whether he had changed the system date on his LAN as a precaution. He told me that he had attempted to follow our advice on the 14th but discovered that the system date was already set to the 15th January. How's that for good luck!

VIRUS ANALYSIS

Jim Bates

777 - Revenge Attacker

Whatever gods concern themselves with electronics in general and computers in particular, must surely have a strange sense of humour. Within normal and reasonable programs, the tiniest mistake is almost guaranteed to cause havoc - yet within the nastiest of virus code, mistakes of legendary proportions gather in multitudes and still the code will spread and destroy! Or perhaps I am being unfair in attributing such perversity to the gods - maybe this is a case of the Devil looking after his own. Certainly the virus under examination this month bears so many mistakes that it should be easy to spot well before any irreparable damage has been caused.

Background Information

The virus is called 777 - Revenge Attacker because this text is contained within the code. The earliest report of it came from the Philippines in mid-1991. It has recently surfaced in Europe but even though the trigger routine is highly destructive, the virus is unlikely to cause too many problems since it does not live happily in any DOS environment.

Structure

Revenge Attacker is a parasitic infector which appends its code to COM files only (including COMMAND.COM). The code becomes permanently resident in memory but there are no stealth routines and so the additional 1127 bytes of virus will be noticeable from a normal directory listing.

While the virus code circumvents the READ ONLY attribute capability, it makes no effort to preserve a file's date and time setting during infection. Thus any infected file will show the date and time of infection in its directory entry.

No encryption is used and so the text mentioned above can plainly be seen when examining the file with an editor. So far, there is little to attract any interest - this is much like most of the other childish efforts that I see everyday. However, two things make this code just that little bit different: the first is the sheer number of coding errors and the second is a ludicrously silly multiple installation routine, which the code uses to determine when to invoke the destructive trigger.

Installation

An infected file has its first six bytes overwritten by a three byte jump followed by the number '777' in ASCII characters. These are used as a self- recognition indicator by the virus to avoid re-infection. When an infected file is executed, processing passes directly into the virus code via the initial jump and the first routine issues an 'Are you there?' call to the system by placing a value of 0FFFFH into the AX register, a value of zero into the CX register and then issuing an INT 21H request. If the virus is resident in memory, the handler will add 2 to the value in the CX register before returning the request. The calling routine now checks the CX register for a value less than 6 and if found, processing is passed into the installation routine proper. At this point the above will seem to make no sense but bear with me - all will be revealed!

The installation routine first checks its internal buffer to see whether the host program originally began with a zero byte. This only makes sense as an aid to getting the original virus code into memory (before it became attached to a file).

Under normal operation, processing then moves the virus code up to offset 103H in memory. Remember that this virus infects only COM files and therefore functions in a single segment with a standard Program Segment Prefix.

Once the code has been relocated, processing returns into it and continues by hooking the INT 21H vector and installing the virus' own Handler. Note that the handler is installed by the legal DOS functions GETVECTOR (35H) and SETVECTOR (25H) and becomes part of the interrupt chain. This is important since later operations will install other copies of the virus and the additional handlers in the chain will all contribute their addition of 2 to the CX register. In this way, the check for a value of less than 6 in CX makes perfect sense as a means of counting how many times the virus code has been installed. (The reason for this process is explained in the section on the trigger routine, below.)

The installation routine continues by freeing as much memory as it can without compromising its own existence, and then setting up a parameter block prior to calling an undocumented function of DOS which loads and relocates a file up to, but not including, actual execution. It is now clear why memory was cleared and once this function has been invoked, the virus routine inserts the original six bytes that were at the start of the host file and then transfers control into the host. The environment of the newly loaded host has a termination address inserted to ensure that when execution finishes, processing returns into the virus code. Once processing does return (the return code is preserved), the virus cleans up the memory allocation, calculates how much memory it requires (1648 bytes) and finally exits back to the system through the normal DOS Terminate and Stay Resident function (31H).

Resident Effects

On initial infection only a single INT 21H Handler routine is installed. This has two functions - firstly it must answer the 'Are you there?' call (and increment CX) and secondly it will intercept certain function requests. There are just two functions intercepted and these are, unusually, functions 3BH - called to change directories, and 0EH - called to change drives. Other function requests pass unhindered.

The interception processing is the same for both functions and it is here that the virus causes some interesting effects.

The handling routine for the interception is itself part of the system once it becomes installed and hooked into the chain. The virus first completes the original request under its own control and then issues its own series of requests in order to locate files suitable for infection. Although the writer attempts to create and maintain his own area of operations for this, the interaction between what the virus wants to do and what DOS is expecting causes a conflict.

The actual infection search and check routine works well but only in the current default directory. As each COM file is found, the first six bytes are read into memory and a check is made for the '777' signature in bytes 4,5 and 6. If found, the file is already infected and so the search continues. Otherwise, the first six bytes are stored within the virus code, the new six bytes (three byte jump and '777') are calculated and written to the beginning of the file and the virus is appended to the end.

Finally the file is closed and processing returns to the caller. Only one file is infected during each intercepted function request and once all COM files in the current directory are infected, no further infections occur until the directory is changed. It should be noted that no check is made on the size of the target file so it is probable that infection of files above 64408 bytes will cause system malfunction.

Side Effects

While this virus is resident, various strange effects may be seen when issuing commands that use the intercepted functions, such as DIR or CHDIR. It is not possible to predict the exact effects on show since these will vary depending upon the version of the operating system, other resident software and even hardware configuration.

However, the commonest effect occurs during a simple DIR command and results in the first file (or the first COM file) being shown repeatedly for every entry that should have been seen. This repetition of filenames is a sure giveaway but may not occur under some cache systems. Other effects include continually repeating characters and/or digits on the screen or the machine may just hang when a DIR (or related) command is issued.

Trigger Conditions

As mentioned above, each time an infected file is executed it will install its own copy of the virus into memory regardless of any other copies that may already be resident. The consequent chaining of the INT 21H handlers will eventually result in the success of the original check of the value in the CX register after the 'Are you there?' call. In the sample analysed, this check would succeed after three infection installations, but it should be noted that the code could easily be altered in this respect.

Once the CX check succeeds, the trigger routine commences and this first uses the BIOS to set video mode 3 (25 rows of 80 columns - text display) and then installs a new keyboard interrupt handler. No attempt is made to preserve the existing handler address and the virus code at this point appears to be a corrupted copy of an original handler which has been deliberately disabled to prevent any keyboard input. Once this handler is installed (via the DOS SETVECTOR call), the following text is displayed:

```
*** 777 - Revenge Attacker V1.01 ***
```

A couple of lines below the text in column one, the first of a series of figure 7s are displayed. As each figure 7 is displayed, the trigger routine writes seven sectors of garbage to logical drives C and D in turn. The routine uses the INT 26H Absolute Logical Write service and the writing begins at logical sector 0 on both drives. This routine goes into an infinite loop and will trash the whole of drives C and D if left to run.

Detection and Removal

The virus employs no encryption and can be detected by the following hexadecimal search pattern:

```
Revenge  Attacker 7510 4080 3F00 750A 4080 3F00
                  7504 F8E9
```

Disinfection is best achieved by locating all infected COM files using a reliable scanner, deleting them using the DOS DEL command and restoring the programs from write-protected master software. In order to ensure that infected files are not intentionally 'resuscitated' using *The Norton Utilities* or similar disk editors, the infected files should be positively erased by multiple overwriting. There are a number of positive erasure utilities available on the market.

777 - Revenge Attacker

Virus Name - 777 Revenge Attacker, no known aliases.

Infects - COM files only (including COMMAND.COM) any size.

Infective Length - 1127 bytes (every time)

Avoids - Files with '777' as the 4th, 5th and 6th bytes

Stays Resident - hooks INT 21H repeatedly, takes 1648 bytes with each installation

Self Check - put 0FFFFH into AX and call INT 21H - CX gains 2 if virus is resident

Trigger - occurs after three infected files have been run, displays message and then trashes drives C and D via INT 26H

SUPPORT SURVEY

Writer: Mark Hamilton
Researcher: Janette Keogh

Out-of-Hours Technical Support

Of the anti-virus vendors contacted, only two provide adequate out-of-hours technical support. This startling conclusion arises out of a survey *Virus Bulletin* conducted during the third week of February 1992 in which the principal suppliers were contacted.

An independent market researcher posed as a secretary working for a fictitious company calling itself *Safka Associates* (actually the name of the researcher's dog!). She was given the telephone numbers published in the documentation of each vendor's package and asked to contact them during the night. Each vendor was asked the following question:

> *'I have just completed scanning my department's computer using your virus scanner. It says my hard disk is infected with the Michelangelo virus. I have valuable medical records I need to protect, what should I do?'*

All vendors should be capable of providing assistance during conventional business hours, but not all users work a convenient nine-to-five day. Some organisations, such as the emergency services, work through the night. Therefore, *VB* wanted to ascertain the various companies' responses to a help call made outside normal business hours.

Suppliers were called between midnight and 2.30am (GMT) on the nights of 17/18th and 18/19th February. Calls to US-based companies were timed to ensure that it was later than 5.00pm local time. The Michelangelo virus was specifically chosen because of its topicality, and because it is a serious and imminent threat to computer users.

Total Control (UK) Limited

0533 883490 (*Bates Associates*)
0483 685299 (*Total Control*)

Time called: Midnight 17/18th February.

Total Control were contacted first. The telephone was not answered after twenty rings. The researcher then contacted *Bates Associates* directly. The phone was answered on the fourteenth ring by Jim Bates who explained to our researcher that 24-hour technical support is not available and that she should contact *Total Control* for technical support in the morning.

Sophos Limited (UK)

0235 559933

Time called: 12.16am 18th February.

Outside normal office hours, *Sophos* has an answerphone which provides the telephone number of the duty technical support person. *Sophos'* Richard Jacobs answered the phone on the fifth ring. He said, 'This is a highly destructive virus and you need to get rid of it'. He then talked our researcher through the correct procedure for removing Michelangelo using the *Sophos Utilities* (provided with *Sweep*). The researcher remarked that he 'was extremely helpful and did not seem to mind about the time of my call'.

Symantec UK Limited (UK)

0628 776343

Time called: 12.40am 19th February.

The call was answered on the second ring by a machine which said, 'The offices are open from 9.00am to 5.30pm. Hold the line and your call will ring throughout the offices; alternatively, if you wish to leave a message in the company's mailbox, please do so after the tone.' The call went unanswered. *Symantec* does not publish the US-parent's phone numbers in its documentation, so no further calls to it were possible.

Central Point Software Limited (UK)

081 569 3316

Time called: 12.45am 19th February.

The phone was not answered after twenty rings.

PC Enhancements Limited (UK)

0707 59016

Time called: 12.50am 19th February.

An answering machine responded on the fourth ring and invited our researcher to leave her name and telephone number.

Fifth Generation Systems (UK)

0494 442224

Time called: 12.55am 19th February.

The phone was not answered after twenty rings.

International Data Security Limited (UK Agent for McAfee Associates)

071 631 0548

Time called: 12.59am 19th February.

The phone was not answered after twenty rings.

S&S International (UK)

0442 877877

Time called: 1.10am 19th February.

S&S's telephone answering machine provides two emergency numbers, the first of which the researcher called and spoke directly to Dr. Alan Solomon. He instructed her to remove the infection using *CleanPart* (a constituent of his *Anti-Virus Toolkit*), having first booted from a clean DOS diskette. He suggested checking all other machines and floppy disks at the researcher's premises and said, 'Don't forget the deadline for clearing the virus is March 6th.' The researcher noted that 'he didn't seem to mind my asking dumb questions at one in the morning!'

VB Software (Eire)

010 353 627 5404

Time called: 1.22am 19th February.

VB Software is the UK/Eire sales and support office for *Leprechaun Software's Virus Buster* package. The phone was answered on the fourth ring by Alan Lowe. He asked for the package's serial number from the master disk which our researcher was unable to provide - 'The master disks are locked-away in the safe.' He then asked for the company name and went off to check the details. After a few minutes, Lowe returned to the phone and suggested that the researcher rename the file '...which neutralises the virus. Get someone in the morning to copy the file so then no one else can run it. Copy that file onto a disk and send it to me in Ireland. I will then send it to Australia - they have to have a live disk in order to produce a cure, as in a medical virus. They will then identify the virus from the disk and determine which version of the virus it is'. He added that a cure should be forthcoming within a week or so.

This response is, if anything, worse than no response at all. It would have been barely acceptable had a parasitic virus infection been reported. Quite how the researcher was expected to rename a file when Michelangelo is a boot sector virus was not explained. Had this been a genuine infection, finding, renaming and copying a non-existent file (as instructed) would have been guaranteed to cause confusion.

Microcom (France)

010 331 46 62 68 68

Time called: 1.45am 19th February.

The phone was not answered after twenty rings.

RG Software (US)

0101 602 423 8000

Time called: 1.50am 19th February
(5.50pm local time 18th February)

An answering machine picked up the line on the sixth ring and invited the researcher to leave a message.

Xtree Company (US)

0101 805 541 0604

Time called: 2.00 am 19th February
(6.00pm local time 18th February)

Xtree does not have a technical support capability in the UK or Europe and so a call to the US is necessary. The phone was answered by a machine which stated that 'regular business hours are 8.00am to 5.00pm'.

Fifth Generation Systems (US)

0101 504 291 7221

Time called: 2.05am 19th February
(7.05pm local time 18th February)

The phone was answered on the first ring by a machine which states that the company operates Monday to Friday from 7am to 5pm Central Time. The message also provides a toll-free number for US and Canadian users.

Central Point Software (US)

0101 503 690 8080

Time called: 2.10am 19th February
(6.10pm local time 18th February)

The phone was answered on the first ring by a machine which said, 'Thank you for calling Central Point Software. Our normal business hours are 7.30am to 5.00pm West Coast Time. The technical support division is available Monday through Friday between 6.00am and 5.00pm.'

McAfee Associates (US)

0101 408 988 3832

Time called: 2.20am 19th February
(6.20pm local time 18th February)

The phone was unanswered after twenty rings.

Microcom (US)

0101 919 490 1277

Time called: 2.25am 19th February
(8.20pm local time 18th February)

The machine, which answered on the first ring, informed the researcher that the company's business hours were between 9.00am and 6.00pm Monday to Thursday and 9.00am and 5.00pm on Friday.

Limitations and Bias

Inevitably, the researcher could not call all suppliers of anti-virus software because some do not provide contact telephone numbers. These include *IBM*, *Frisk Software* and *ESaSS*. Similarly, she was unable to contact *Symantec's* US offices as the company's documentation only provides UK contact details. The tests were biased in favour of US-based suppliers by virtue of the relative times the calls were placed - yet none of the US suppliers provided the researcher with anything more substantial than a pre-recorded message.

Plaudits and Brickbats

Generally, this test produced depressing and often dismal results - software developers are only too happy to sell their products worldwide: in their eagerness to export, anti-virus software developers should remember that computer viruses do not respect time-zones!

Plaudits to both *S&S* and *Sophos* for providing technical support in the middle of the night and brickbats to *VB Software* for providing erroneous and confusing instructions. Of the three companies which responded, only *VB Software* made any attempt to verify software registration. *Sophos* provided technical assistance first and then asked for our researcher's name and organisation; *S&S* didn't request any registration or end-user information at all. Interestingly, neither Sophos or *S&S* asked our researcher whether she would be reporting the incident to *New Scotland Yard's Computer Crime Unit* nor did either company mention the fact that a criminal offence covered by *Section 3* of *The Computer Misuse Act* 1990 had occurred. Both companies gave such an undertaking at last year's meeting of the *Computer Virus Strategy Group* hosted by *New Scotland Yard*.

Editor's Statement

This survey was conducted with the prior knowledge of *the editor*, *the author of this report* and *the researcher* who conducted the survey. No other person had prior knowledge that this survey was being conducted.

BOOK REVIEW

Richard Jacobs

The Little Black Book of Computer Viruses

The Little Black Book of Computer Viruses claims to be the first in a series of three books on computer viruses. This first volume is described as 'a technical introduction to the basics of virus writing.' Apparently the later volumes will cover scientific and military applications for computer viruses.

Following a ludicrously weak attempt to justify virus writing and thus legitimise this book (accompanied predictably enough by an obligatory dig at the US government), Mr. Mark Ludwig confronts the technicalities of computer viruses. He starts with a clear and concise description of the basic elements of a computer virus. The rest of the book describes four simple viruses; two parasitic and two boot sector. Each virus is described in detail, explaining why each element is necessary and how the different parts are put together. Interestingly, the copyright notice for the book states that it was written in 1990 - did Mr. Ludwig have difficulties in finding a publisher for this explicit treatise?

The author displays considerable knowledge of the internal structures and working of both DOS and the system BIOS, although there is no description of the many undocumented features seen in recent viruses. He appears confused regarding the purpose and target audience of his book. He correctly states that most viruses are written in assembly language and so some knowledge of assembly language programming is a prerequisite for virus writing. Clearly, the only way that anyone could 'benefit' from virus writing is to understand what they are doing so that they can take existing ideas and develop them. It is therefore somewhat surprising to find this book describing the number and purpose of 8088 registers, along with a description of how to calculate absolute memory addresses. Anybody who was unfamiliar with such basics would not be capable of programming in assembly language. Rather than learning from the exercise of writing these viruses such a reader could only type them in and release them. This contradicts the claimed justification of the book. This contradiction is taken a step further in a move that simply cannot be justified: the viruses are not only supplied in assembly language form, but also as hex dumps, that can be loaded straight onto a PC in executable form. The sole purpose of these dumps is to enable people who do not have assemblers to run the viruses. Such people cannot possibly learn anything from entering unintelligible sequences of numbers and following instructions to execute them!

Neither of the two parasitic viruses have any memory-resident element. The first, called 'Timid', simply searches the current directory for .COM files and infects the first one it finds, that has not already been infected. If it does not find one, the virus simply returns control to the host program. The various functions of the virus are described in detail along with the relevant DOS structures and processes. Despite perpetual protestations that the object of this book is to further freedom of thought and research, the author indicates the location at which a destructive routine could be added to the virus, even calling a function 'DESTROY' in the example. The 'DESTROY' function is not included, but any semi-competent assembly language programmer could add it.

The second parasitic virus is called 'Intruder' and is considerably more complicated than the Timid virus, although it still does not go memory-resident. This virus only infects .EXE files and searches through sub-directories in addition to the current directory for files to infect. Intruder includes two 'anti-detection' measures. The first of these is the almost universal move to ensure that it does not change file attributes and date/time. The second approach, although very simple, is far less common. The virus does not infect a file every time that an infected file is run. This measure could be developed to create problems with detection for users relying on checksumming software, who are not aware of this possibility.

The third virus is a simple DOS boot sector virus that infects other disks, but has no other function. There are two unusual aspects to this virus. The first is that the virus does not make itself memory-resident and so can only infect during the boot process. This greatly reduces the chances of the virus spreading rapidly as the only way that it can infect a new floppy disk is if there is a disk in drive B: when the PC is rebooted.

"In a move that simply cannot be justified, the viruses are not only supplied in assembly language form, but also as hex dumps, that can be loaded straight onto a PC in executable form."

Unlike most existing boot sector viruses this virus does not copy the original boot sector. Instead the virus reduces the normal DOS boot sector code to the absolute minimum required to boot the PC and incorporates the entire contents of its own code within the space created. The advantage of this tactic is that the new boot sector functions as a virus but also as an active DOS boot sector and so no other disk space is required.

The final virus, 'Stealth', is a more conventional boot sector virus which demonstrates various techniques for storing the extra sectors required by most boot sector viruses.

On 5.25 inch disks the virus formats an extra track and stores its code on it. On 3.5 inch floppy disk formats the virus stores its code in the final sectors of the disk and marks the relevant clusters as BAD. On hard disks the virus uses the unused sectors of the first track to store its extra code. The virus infects hard disks at boot time and floppy disks every time their boot sector is read, so unlike the other viruses in this book it does go memory-resident. The 'Stealth' virus contains simple measures to conceal its presence. Attempts to read/write the infected boot sector are redirected to the original clean boot sector. Likewise, attempts to read other virus contaminated sectors of the hard disk are intercepted.

The Little Black Book of Computer Viruses is an irresponsible and potentially harmful publication. The source code (ASM, PAS and BAS files), the hex listings and compiled, executable programs for all the viruses and related programs in the book are supplied by the publisher on disk! The disk ($15.00) is, *of course*, for information purposes only. Mr. Ludwig will doubtless take cover beneath the *Fifth Amendment*. Coming from a country where gun control is virtually non-existent, this book might be regarded as *relatively* innocuous - a fact which will be of little comfort to afflicted computer users.

'BLACK BOOK' VIRUSES

Detection patterns for IBM PC viruses published in source code form in *The Little Black Book of Computer Viruses*.

LBBCV-Intruder - EN: Trivial virus published in *The Little Black Book of Computer Viruses*. No side effects.

```
LBBCV-Intruder E867 0375 18E8 6B03 E86E 03E8
               2600 7509 E891 03E8 E401 E8CE
```

LBBCV-Kilroy - DN: Trivial virus published in *The Little Black Book of Computer Viruses* . No side effects.

```
LBBCV-Kilroy   721A 813E FE06 55AA 7512 E8FE
               00BA 8001 B901 00B8 0103 CD13
```

LBBCV-Stealth - MR: Trivial virus published in *The Little Black Book of Computer Viruses*. No side effects.

```
LBBCV-Stealth  FB80 FC02 740A 80FC 0374 3C2E
               FF2E 3070 80FE 0075 F680 FD01
```

LBBCV-Timid - CN: Trivial virus published in *The Little Black Book of Computer Viruses*. No side effects.

```
LBBCV-Timid    2EFC FF09 00BA 2AFF B41A CD21
               E83E 0075 10E8 8F00 BA48 FFC7
```

Title: *The Little Black Book of Computer Viruses* (178 pp.)
ISBN: 0-929408-02-0
Author: Mark A. Ludwig
Publisher: *American Eagle Publications, Inc.*, PO Box 41401, Tucson, Arizona 85717, USA.
RRP: $14.95

BOOK REVIEW

A Pathology of Computer Viruses

Rarely does *VB* unequivocally recommend anything other than sound backups and a healthy degree of scepticism. It is refreshing, therefore, to describe *A Pathology of Computer Viruses* as a *Meisterwerk* - quite simply, the most comprehensive published account of malicious software on PC, Macintosh and Unix platforms currently in print.

This sound treatise covers the history, theory and functioning of viruses, Trojans, worms and other forms of malicious software and provides a thorough introduction to virus control. Not necessarily a book for the squeamish, the sections on virus mechanics and propagation are quite explicit but the author, unlike the Burgers or Ludwigs of this world (see book review, page 17), is never vulgar - source code and sensitive programming tactics are conspicuous by their absence. It is impossible to publish any useful information about viruses without revealing something of the methods by which these programs work. In this instance, the author reveals information of genuine use to those engaged in combating the virus threat and, at the same time, adeptly avoids crossing the boundaries of good taste. It is also nice to see a book on computer viruses which does not contain self-righteous and mealy-mouthed statements of either an apologistic or condemnatory nature - the author remains completely dispassionate throughout and never resorts to the politics of the soap-box.

The narrative style of the book is mature, detached and analytical - this is a scholarly work devoid of the sensationalism, alarmism, self-aggrandisement or commercialism which has been the hallmark of certain other authors in this field. Mr. Ferbrache's qualifications for writing this book are impeccable. As a research associate at *Heriot Watt University*, Edinburgh, he spent some four years studying threats to computer integrity; as a result he is now a recognised authority on both Macintosh and Unix security.

Ferbrache has no axes to grind - having eschewed the world of anti-virus product development for a career with the UK's *Defence Research Agency* he can tell the story objectively and without bias. The references section of the book is testament to this fact, some seven pages of citations list virtually every major book, magazine, article, refereed paper, advisory, BBS forum or other document of value (only Professor Lance Hoffman's excellent compilation *Rogue Programs: Viruses, Worms, and Trojan Horses* and the various studies of the *National Computer Security Association* are missing).

The book provides a strong historical perspective outlining events both trivial and monumental. The author describes developments from the mainframe 'rabbit' programs of the 1960s to the proliferation of viruses on PCs worldwide. Early *Xerox* experiments at Palo Alto, Dr. Cohen's revolutionary

papers, the Internet worm, the AIDS Trojan, the establishment of *CERT*, and the growth of the anti-virus software industry (not to mention the birth of *Virus Bulletin*) are amongst the more momentous events described. Minutiae such as the publication of the influential sci-fi novel *Shockwave Rider* (which contained a rogue computer program), the establishment of the quixotic *British Computer Virus Research Centre* and even the use of a virus-like program on TV's *Star Trek* are thrown in for good measure.

Theory is discussed but not over-emphasised. One chapter of the book is devoted to the analysis of biological viruses and this provides a useful insight into a world into which computer virologists rarely wander. From then on, the hard practicalities of life predominate. Successive chapters deal with PC viruses (including infection mechanisms, camouflage and trigger routines), virus prevention, detection and removal, the Apple Macintosh virus families, mainframes and Unix systems, network attacks (the Internet worm, the WANK worm, CHRISTMA EXEC *et al.*) and the international response to these attacks. Explanations are very thorough with diagrams where relevant and each topic is examined in considerable depth. The author shows a remarkable grasp across a range of operating systems and architectures and neither sidesteps nor ignores complex technical issues for the sake of convenience or brevity.

Criticisms are few. The book does suffer from a slight British parochialism, particularly with regard to its concentration on the UK's *Computer Misuse Act* and relative disregard for equally important US and European legislation. Similarly, United States and 'rest of world' anti-virus product manufacturers and virus control agencies are occasionally overlooked - John McAfee hardly gets a mention, while the *Computer Virus Industry Association* and the *National Computer Security Association* are completely ignored. There are also some nasty typographic errors which will irritate the proof-readers among you but given the scope of this book and its highly technical content these errors are perhaps forgiveable.

In summary, Mr. Ferbrache provides the most complete picture of the virus and 'malware' phenomenon to date. Undeniably an ambitious book, *A Pathology of Computer Viruses* succeeds in virtually all of its objectives. This is not a do-it-yourself defence manual (no free software, discounts or other gimmicks are included with purchase) but a cold-blooded, authoritative and thorough examination of a modern day problem. Wholeheartedly recommended.

Title: *A Pathology of Computer Viruses* (299 pp.)
ISBN: 3-540-19610-2 (UK)
0-387-19610-2 (USA)
Author: David Ferbrache
Publisher: *Springer Verlag*
RRP: £24.95

PRODUCT REVIEW

Dr. Keith Jackson

PC Armour

S&S International's PC Armour aims to give users control over unauthorised access, data theft, viruses, Trojan horses and pirated software. It achieves this by controlling who can gain access to a PC, permitting only the execution of authorised programs, and preventing anyone from accessing the hard disk if the PC is booted from a floppy disk. This review will cover each of these three aspects individually.

Although *PC Armour* works with a network, I had no means of testing this feature, and this review refers only to standalone operation. *PC Armour* is a generic protection program; therefore, testing with a multitude of virus infected files is pointless. Instead, it is necessary to establish whether the principles and implementation of the program are sufficiently well conceived to provide protection against viruses and other security threats. *PC Armour* requires a memory-resident portion to be present when it is executing, but as this occupies less than 2K of RAM, it should not have a major impact on total available memory. The manual states that program authorisation does not work with *Microsoft Windows*.

Presentation

I have been given Alan Solomon's products for review by various publications over the past few years, and I have to admit that the overall presentation, documentation and packaging has now improved by leaps and bounds. *PC Armour* comes as a boxed A5 ring-bound manual, which contains an instruction manual (22 pages), and a *Data Security Handbook* (112 pages) which provides a clear explanation of the underlying rationale behind products such as *PC Armour*. Both are well written, pleasingly laid out, and thoroughly indexed. As you would expect from Alan Solomon, the *Data Security Handbook* excels in its virus coverage but it also covers other aspects of PC security reasonably well. The software is provided on both 3.5 inch and 5.25 inch diskettes.

Getting Started

The *PC Armour* installation program is simple to use, permits *PC Armour* security (boot protection, program authorisation and access control) to be installed in any combination and provides on-screen help for all of its features. Before proceeding, the installation program asks for confirmation that the hard disk has been completely backed up; a wise precaution.

The manual states that *PC Armour* must be installed onto the first bootable hard disk drive. Given that *PC Armour* enforces boot protection, it is difficult to see how this constraint could

be removed. The files used by *PC Armour* all reside in the root directory. I have an aversion to programs that clutter up the root directory, however, with the exception of the authorisation program, these are at least hidden files.

I encountered no problems installing or de-installing *PC Armour*. This is an understated compliment, as I have quite a complicated boot process which allows me dynamically to choose one of four different methods of booting. This did not perturb *PC Armour*. At first glance I was somewhat perplexed to find that most of my AUTOEXEC.BAT file did not execute after *PC Armour* had been installed, but this was just the authorisation feature doing its job. The default installation authorises only the authorisation program, the password checking program, and COMMAND.COM (the MS-DOS command interpreter). All other programs (which includes programs executed from within AUTOEXEC.BAT) must be explicitly authorised by the user. Note that if you use an alternative command interpreter (such as *4DOS*), then this must be authorised manually.

Given the number of security products that I test, I was especially pleased to see decent de-installation utilities being provided. Many times in recent years I have encountered serious problems when trying to remove a security 'feature' after I've finished reviewing it. The worst so far had better remain nameless - it fouled up my hard disk so thoroughly that it had to be returned to the disk vendor as it prevented the PC from even booting.

One minor point, the last part of the de-installation finishes by putting the message '*PC Armour* completely removed' on the screen. This is not quite true, as a copy of the authorisation program is left behind in the root directory of the hard disk. I'm aware that the manual advises that this program should be copied to floppy disk and stored in a secure place, but it is very easy to forget to erase the root directory copy.

Passwords

PC Armour can either use individual passwords to control its separate features (the most secure option), or it can make the passwords the same (the easiest option to remember). All passwords must be at least eight characters long, with no distinction being made between upper case and lower case letters. I like this; case sensitive passwords are a nuisance, as it is very easy to forget the exact location of the capital letters. The eight character minimum length ensures that guessing a reasonably random password is to all intents and purposes impossible.

PC Armour passwords are stored within the authorisation program, but they appear to be encrypted. I could find the location in this file which altered when the password was changed, but I could not relate the altered bytes to the password. Someone attempting to discover a password would need to reverse-engineer this encryption. My own feeble attempts failed to achieve this.

If you do forget a password, then the developers of *PC Armour* can help you out. Inevitably this means that there is a back door through *PC Armour*, but given that the alternative could well be an extremely irate user with hundreds of 'lost' files, and given that *PC Armour* does not pretend to offer 'high' security, this is probably justified.

Boot Protection

I find it difficult to find anything to say about the boot protection offered by *PC Armour*. It works! If you boot from floppy disk and try to access a program on the hard disk, then MS-DOS simply reports access denied. Note that this error is not reported by *PC Armour*, but by MS-DOS itself.

Given that *PC Armour* operates without any additional hardware, then it must be changing something in the low level structure of my hard disk which prevents it being recognised by MS-DOS when the PC is booted from a floppy disk. I don't know what these changes are, and even if they were explained to me, *Virus Bulletin* is hardly the place to publish such details when they could be of benefit to anyone (or anything, perhaps a virus) trying to circumvent the boot protection.

Program Authorisation

Unsurprisingly, the documentation provided with *PC Armour* does not explain exactly how program authorisation is carried out, but it soon becomes obvious that when a program is authorised, its time stamp changes. This was the only evidence that I could find that a program was authorised by *PC Armour* to execute. The time stamp inserted by *PC Armour* does not always make sense as far as the twenty four hour clock is concerned, for instance I saw a timestamp of 30 hours 42 minutes attached to one file, when normally anything beyond 23 hours 59 minutes is classed as an error.

Using the time stamp on a file as a means of authorising program execution has the odd side effect that some files appear to move into the future. For instance I first installed *PC Armour* early one morning, yet the installed *PC Armour* authorisation program had the correct date stamp, and a time stamp set to just after half past four in the afternoon. This totally confused *Norton Commander*, a well-known MS-DOS shell, and made it believe that the authorisation program was older than all other programs in that particular subdirectory. Curious.

As the content of an executable file does not appear to be changed by the authorisation process. I can only conclude that the authorisation process really does only set the timestamp to indicate that the program has been authorised with a particular password. *PC Armour* must then check this authorisation at run time. It is therefore inevitable that there is an overhead associated with having program authorisation installed. I measured the time taken to load several MS-DOS programs, with and without *PC Armour*. The results of these tests are shown in *Table 1*.

Program	Function	Normal Execution (seconds)	With PC Armour (seconds)	% Increase
Manifest	Utility	2.9	5.3	83
LapLink	Data transfer	3.2	5.0	87
Hog	Graphics	1.2	2.4	100
Aseasyas	Lotus 123 clone	3.0	7.8	160
Procomm	Comms package	1.9	6.2	226

Table 1. Load overhead imposed by *PC Armour*

In short activating the program authorisation feature roughly doubles program load time. My first attempt at measuring these timings produced changing, inconsistent, results, until I realised that this was caused by the presence of the MS-DOS disk cache utility SMARTDRV. When the disk cache was removed things became consistent. I attempted to measure the effect of program authorisation on the speed of program execution (excluding any effect due to program loading). The results were inconclusive, even down to the MS-DOS utility CHKDSK consistently running 20% faster when *PC Armour* was active, which mystifies me completely. I surmise that the effect on actual program execution time is close to zero.

I encountered a problem whenever I tried to execute an MS-DOS program from within another program, a process known as 'shelling out'. Whenever I attempted this, an 'Access denied' error message was reported, and I was unable to return to the original calling program. This problem must be caused by *PC Armour* interfering with the execution of a new copy of COMMAND.COM after shelling out, but I'm not quite sure why. I don't keep COMMAND.COM in the root directory, but surely *PC Armour* should cope with this?

Access Control

PC Armour's access control facility is activated by a program executed from within the MS-DOS batch file AUTOEXEC.BAT. The installation program alters the first few lines of AUTOEXEC.BAT to activate this feature before any other programs are executed at boot time. It also makes AUTOEXEC.BAT a hidden file to prevent idle tampering. The access control feature requires that the correct password is entered before the PC is allowed to boot.

Summary

The manual makes it clear that none of the security features that *PC Armour* offers are 'high security', and rightly points out that a high level of security can only really be achieved by adding a plug-in card to the PC. Such a card may well be five to ten times more expensive.

One charge that can be levelled against *PC Armour* is that some of its usefulness could be removed if the MS-DOS AUTOEXEC.BAT file was altered. This is true. However if the access control and boot protection features are both active, this can only be achieved by knowing the correct password. Therefore although the constituent parts of *PC Armour* can be activated individually, it is best to use all three component parts together.

De-installing *PC Armour* does not remove the eccentric time stamps from the 'authorised' programs. This is inevitable. If the time stamps were to be restored to their original value, then *PC Armour* would have to maintain a list of all files that had been authorised, their original time stamps, and where the file is located. No doubt some fool would then delete this list, or the executable files could be moved. It is therefore a direct consequence of the *PC Armour* program authorisation method that some information is left behind.

I liked *PC Armour.* Unlike so many products which *VB* has reviewed recently, this program is well conceived, well documented and well implemented [*Steady on, Dr. Jackson - we don't want to inflate these peoples' egos! Ed.*]. Although program authorisation imposes an inevitable overhead on program loading, using disk cache software can reduce this overhead almost to zero. I routinely use a disk cache, so the program loading overhead does not worry me unduly (it's inevitable anyway), and the features offered by *PC Armour* are worthwhile outside a high-security environment.

Finally...

Be sure to obey the manual when it says that you should possess a bootable floppy disk; *PC Armour* cannot be de-installed without first booting from a system diskette. The documentation makes this crystal clear, but who reads a manual before installing software?

Technical Details

Product: *PC Armour*

Developer/Vendor: S&S International Ltd., Berkley Court, Mill Street, Berkhampstead, Hertfordshire HP4 2HB, Tel: +44 (442) 877877, Fax: +44 (442) 877882.

Availability: Any IBM PC, XT, AT, PS/2 or compatible personal computer, either standalone or on a network.

Version Evaluated: 1.10.00

Serial Number: None visible

Price: £69 sterling plus VAT, one-off. Site licences are available.

Hardware Used: Toshiba 3100SX laptop with a 16MHz 80386 processor, 40Mbyte hard disk, 1.44Mbyte floppy disk drive, running under version 5.0 of MS-DOS.

PUBLIC DOMAIN

Mark Hamilton

Stealth Bomber

In the main, the response to the virus threat has been reactive as opposed to proactive i.e the anti-virus industry has been developing programs which detect viral activity after infection. With few exceptions, the software industry has done little to protect its products from virus contamination and yet, as Kevin Dean, developer of *Stealth Bomber* (the subject of this review) has proved, self-protection can be simple to implement and reasonably effective. To put *Stealth Bomber* in context, it is worth summarising the various ways in which files can be checked for virus activity. First, there is the virus-specific scanner which has the advantage of being able to positively identifing a range of known viruses. The major disadvantage is that scanners have to be frequently updated with recognition patterns as new viruses appear.

Then there are generic checkers which calculate checksums and, in some cases, save essential information about executable programs in an external file. Should disaster befall the user, the same generic checker can attempt to repair the infected file. As a long-term defensive strategy, generic checkers play an important role but their major disadvantage is that the original source of infection is not identified - only its effects. As an alternative to placing the checksum information in an external database, at least one anti-virus developer favours appending this information to the end of the executable file. This practice is frowned on by many specialists who argue that the pure program code is sacrosanct and shouldn't be altered in any way. *Central Point Software* took the appending strategy one stage further by producing a program which adds a self-checking 'stub loader' program to the end of protected files. In practice, this form of inoculation is somewhat limited in the types of files that can be protected (see *Virus Bulletin*, June 1991, page 22).

An alternative to generic checking is integrity checking which programs do as part of their own initialisation. It is this type of anti-tampering code which may prove useful to program developers and the software industry in general. Apart from anti-virus software, I have come across only one end-user application which incorporates integrity checking (*Microsoft's Visual Basic*, a *Windows 3* application).

Kevin Dean

Kevin Dean is a Canadian software engineer working on communications systems who says he wrote *Stealth Bomber* as an intellectual exercise. He believes that developers could do much to inform their users that a particular program has been altered by incorporating a simple set of routines such as those contained in his *Stealth Bomber* system.

Stealth Bomber is logically divided into three components: CRCSET.EXE looks for a predetermined ASCII string within an executable file and replaces it with a CRC polynomial and checksum. *CRCSET* is delivered as a compiled program - Dean says he is unwilling to release the source to prevent Trojanised versions appearing and because he is using 'cute programming tricks'.

Unlike other programs which calculate CRCs, *CRCSET* does not use a fixed polynomial; it (along with the CRC itself) is calculated at run-time. Dean says that he has optimised the manner in which the calculations are made so that the time to generate a CRC is more or less constant, regardless of the number of bytes in the file. Both the Polynomial and the CRC are 32 bits wide (long integers) meaning that, for all practical purposes, any change in the file will be detectable.

There are equivalent programs to *CRCSET* but they operate in a different way - they split the file into three physical parts:

| Beginning of File | CRC | Rest of File |

They calculate the CRC on 'Beginning of File', skip 'CRC', and resume calculating the CRC at the 'Rest of File'. Dean maintains this method is inherently flawed because the CRC can be fairly easily located, the file modified and a new CRC calculated to replace the old one. He claims his scheme makes both the CRC and the polynomial part of the data stream itself - the bytes that comprise these components are also read and checked *[?! Ed.]*. According to Dean, this makes it more difficult to introduce undetectable changes to the file.

The other components are delivered as both *Turbo Pascal* and ANSI C source code files. As I am more familiar with C than with Pascal, I tested the C version. The C sources compile with any ANSI compatible compiler such as *Jenson and Partners' TopSpeed C*, *Borland C++*, *Turbo C* and *Microsoft C*. There are no model-specific memory requirements. To use *Stealth Bomber*, the user simply includes its provided sources in his project or *make* file and adds two calls in his main procedure: *stealthsyscheck* and *filesyscheck*. Both return a non-zero result if there's a problem.

Stealth Checking

Dean added mechanisms for checking for stealth viruses into version 2.0 of *Stealth Bomber* having seen at first hand the problems likely to be encountered when dealing with viruses such as 4K (Frodo). Therefore, *Stealth Bomber* is designed to work in a potentially 'dirty' (stealth-infected) environment. The stealth checking he has implemented is somewhat rudimentary, yet effective - in tests, it detected environmental changes caused by the 4K and SVC 6.00 viruses - but may not be quite so effective should virus writers determine mechanisms specifically targeted to evade *Stealth Bomber*.

By walking the DOS memory chain, he calculates the amount of memory below 640K and this is compared by the value the

BIOS returns. If the two agree, all well and good (in fact, under MS-DOS 5 there is a discrepancy of 16 bytes and this is catered for) and the tests proceed. Dean also checks interrupts 21, 24, 25, 26, 1C and 28 (Hexadecimal) for subversion and looks for two types of instructions at the very beginning of the interrupt code which, if present, indicate that the interrupt has been tampered with. The particular instructions are:

```
JMP FAR <address>        or      JMP FAR CS:<address>
CALL FAR <address>       or      CALL FAR CS:<address>
```

There may be TSR programs which quite legally start their interrupt handlers in this way, so checks are also made to see if the target of the jump or call lies within the same Memory Control Block or outside conventional memory (i.e. in an Upper Memory Block or in the High Memory Area) in which case the interrupt redirection is valid.

There is a readily apparent loophole in this test. This self-checking method is not foolproof; suffice it to say that a virus could easily be written which circumvents it. However, Dean's stealth checking works with all currently known viruses (both 'in the wild' and 'lab' samples). However, this may not be the case in the future as more sophisticated viruses appear.

File Checking

Having checked that the environment is 'clean' using the *stealthsyscheck* function call, the programmer then calls the *filesyscheck* and passes it the name of the executable file to be checked (under DOS 3 and above, this can simply be stated as 'argv[0]' or its *Turbo Pascal* equivalent) along with a pointer to the stored CRC/polynomial pair. When writing the code, this is defined as:

```
typedef unsigned long crc32t;
typedef union {
        char searchstr[8];                  /* Used by CRCSET to
                                               locate position */
        struct {
                crc32t polynomial;          /* Polynomial for
                                               file */
                crc32t crc;                 /* Calculated CRC for
                                               file*/
                } x;
        } filecrc;
const filecrc fcrc = {
        '', 'S', 'T', 'E', 'A', 'L, 'T', 'H'
        };                                  /* Default value used by
                                               CRCSET */
```

Thus, the call would be:

```
result = filesyscheck(argv[0], fcrc);
```

As well as checking the CRC, this function checks the file's directory entry and looks for an invalid date/time stamp and to see if the file length in the directory entry agrees with the number of bytes it has read from the file. Again, the result returns a non-zero value if a problem is encountered - like the stored CRC not matching the calculated value. Having compiled and linked the application, *CRCSET* is run and this looks for the default string 'STEALTH' (this can optionally be changed) which it replaces with the polynomial and CRC, thus 'arming' the program.

Caveats

Stealth Bomber is a laudable attempt at producing self-checking code, but there are some caveats which need to be pointed out. You can not compress executable programs before or after running *CRCSET* using utilities such as *PKLITE*, *DIET* or even *EXEPACK* since either the decompressor will fail to work correctly or an incorrect CRC will be generated (because the file has been changed). You cannot use anti-virus programs which modify the executable in some way (such as *Central Point's* inoculator) nor can the program executable be modified either by itself - a common practice among the *Turbo Pascal* fraternity, or by an installation routine to imbed the user's name, for example. Finally, it should be remembered that this self-checking method can be subverted.

Kevin Dean admits to limited virus knowledge and is therefore trying to second-guess the virus writers. I hope that development of this program continues. I also hope that software developers take his ideas on-board and start developing their own self-checking methods. Self checking code is by no means a panacea and should not be solely relied upon, but it does give users a fighting chance against virus contamination and malicious tampering. Despite these benefits, the warning must be repeated; *Stealth Bomber* is effective at the moment - it may not be in the future.

Warning

There is an earlier version of *Stealth Bomber* in circulation, version 2.0. This version produces false positive results when run under DR-DOS 6 or MS-DOS 5. Version 2.2 clears these problems.

Product and Version: *Stealth Bomber* version 2.2

Availability: *CompuServe*: IBM Programmers' Virus Forum

Filename: STEALTH.ZIP

Author: Kevin Dean. *CompuServe* ID: 76336,3114

Address: Fairview Mall PO Box 55074, 1800 Sheppard Avenue East, Willowdale, Ontario, Canada M2J 5B9

Cost: None - Public Domain

END-NOTES & NEWS

The *Second International Virus Bulletin Conference*, Edinburgh, 2-3rd September 1992. **Combating Viruses: Corporate Strategies and Technical Developments**. Information from the delectable Miss Petra Duffield, *Virus Bulletin*, UK. Tel 0235 531889.

The University of Bradford has introduced a **Virus Certification Facility** to evaluate anti-virus software. Not to be confused with evaluators licensed by *CESG*, the *Computer & Electronics Security Group* of *GCHQ*, which together with the *DTI* administers the official *ITSEC* certification scheme. Information on the *Bradford University Virus Certification Centre* is available from Simon Shepherd, *University of Bradford*, Dept. of Electrical Engineering, Bradford, West Yorkshire BD7 1DP. Tel 0274 733466, fax 0274 391521.

The Antivirus Methods Congress (AMC) is yet another 'initiative', this time US based, which has set itself the ambitious task of 'retarding and minimizing the onslaught of malicious code'. More 'jaw-jaw than war-war', the initiative is currently bogged down in a mire of red tape about elections, committees, sub-committees and ballots. Despite this, lots of virus 'superstars' are involved. Information from Dick Lefkon, Tel USA 212 663 2315.

Virus News and Reviews is a planned **new monthly magazine** from the *National Computer Security Association* (*NCSA*). The technical editor will be Dr. Alan Solomon and the bulletin will feature comparative product reviews, graphs and tables, tutorials on virus removal, network recovery, brief descriptions of new viruses, more thorough descriptions of the common viruses, etc. US$395 for an annual subscription. (Does any of this sound familiar to anyone?) Information from *NCSA*, Tel USA 202-364-8252. (The editor has just fainted - fetch a strong brandy).

IBM UK Ltd continues a series of **PC Security/Virus Management/Virus Hands-on workshops**. The next take place in Warwick, 23-25th March 1992. Information from the *IBM Systems Management Services Centre*, Tel 0705 323765.

Sophos **virus workshops** take place on 26th May (introductory) and 27th May 1992 (Advanced)in London. Tel UK 0235 559933.

S&S Consulting Group is holding **seminars** on the virus threat (18-19th March), data recovery (13-14th May), PC support (25-26th March) and PC security (1st April). The venue is Great Missenden, Bucks, UK. Tel 0442 877877.

Disknet 'the leading edge of virus technology' **provides program security**, access security and software auditing for the IBM PC. Information from *Reflex Magnetics Ltd*, UK. Tel 071 372 6666.

DataSure and Bryan Clough of *PC Virus Index* (see *VB*, July 1991, pp. 42-43) is offering **virus consultancy** to clients. Managing Director Israel Kay is reported to have said: 'The conventional consultancies have ripped off clients by charging sometimes more than £2,000 a day for advice.' *DataSure's* prices are dependent on the size of site. According to Kay a small site would not be expected to pay more than £1,000 annually. Clearly not a 'rip-off'.

VIRUS BULLETIN

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

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

Editorial enquiries, subscription enquiries, orders and payments:

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

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

US subscriptions only:

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

Tel 203 431 8720, Fax 203 431 8165