August 1991 ISSN 0956-9979

VIRUS BULLETIN

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

CONTENTS

EDITORIAL 2

TECHNICAL NOTES 3

LETTERS

From *IBM* and *Visionsoft* 5

From *Knoxware* and *Trend Micro-Devices* 7

IBM PC VIRUSES (UPDATE) 8

NOVELL UPDATE

Novell's Analysis of the GP1 Virus 9

NOVELL EXPERIMENTS

Virus Propagation and *NetWare* Security 10

VIRUS ANALYSES

2100 and 'Cracker Jack' the Plagiarist 19

PRODUCT REVIEWS

ProScan 23

Virus Guard 25

END-NOTES & NEWS 28

EDITORIAL

Network Experiments

A large proportion of this month's edition of *VB* is devoted to the propagation of computer viruses on *Novell NetWare* PC networks. The paper *Virus Propagation and NetWare Security*, submitted by Dr. Jan Hruska and Richard Jacobs of *Sophos Ltd*, provides some revealing insights into the interaction between various viruses and network software. *VB* is also indebted to Eric Babcock, *Novell's* software security specialist, both for his efforts in peer-reviewing this paper and for supplying a short report on the GP1 virus (see page 9) which arrives at conclusions rather different from our initial speculations published in June 1991. In the absence of evidence to the contrary, Mr. Babcock's report should be regarded as the definitive functional analysis of the GP1 virus.

Offsets Out

An editorial decision has been taken to discontinue publishing offsets for virus search patterns. It is the technical editor's opinion that offsets should be removed from any virus scanning data despite the fact that this will result in a degradation of scan run-times. The reasons for this are twofold: firstly, the removal of offsets substantially increases the likelihood of detecting virus variants, which are appearing at an exponential rate. Secondly, misinterpretation of offset data by at least one programmer involved in the development of a commercial anti-virus product, resulted in *VB* search patterns being invalidated - the scanner was looking for the right patterns but in the wrong places.

Spanish Telecom, Tequila, 2100

End-users of virus-specific scanning software in the United Kingdom should take note that any memory-resident or non-resident scanning software in use ought reliably to detect the Spanish Telecom, Tequila and 2100 viruses. These viruses are in the wild but despite the fact that one of them (Spanish Telecom) was analysed some eight months ago, only four of the thirteen scanners tested in the July 1991 edition of *VB* detected it. The appearance in April of the Tequila virus, which spread across Europe via an infected shareware master diskette, underlines the need to update scanning software on a very regular basis. In light of the rapidity with which new virus infections can spread and take hold, virus-specific software which is updated less than monthly now appears to be of questionable value.

Plagiarism

An Italian boy calling himself 'Cracker Jack' has claimed responsibility for a number of recent computer viruses, some of which we report in this edition.

The samples themselves do not merit detailed technical reporting but examination has revealed that many of this young man's rather amateurish programming efforts have been copied from virus code developed by the Bulgarian virus writer who calls himself Dark Avenger. This obvious plagiarism has almost certainly occurred due to the mushrooming of the virus 'exchange' Bulletin Board Systems which *VB* reported in May this year.

It would appear that cooperation between virus writers is now at an all-time high - the Bulletin Boards are being used as forums to swap ideas, upload and download object and source code as well as the more popular anti-virus public domain and shareware tools (presumably so that they may be subverted). These virus exchange Bulletin Boards are without doubt the single area of greatest concern to the anti-virus community.

Scotland Yard Arrests '8LGM' Hacking Ring

The *City & Metropolitan Police's Computer Crimes Unit*, in a complex combined operation with *British Telecom*, has arrested the three UK-based members of an international hacking ring known as '8LGM'. The operation, the largest of its type and involving eight regional police forces, was mounted during the early hours of Thursday 27th June 1991.

Officers simultaneously arrested Neil Woods of Oldham, Karl Austin Strickland of Liverpool and Paul Daniel Bedworth of Ilkley, West Yorkshire, and charged them with conspiracy to contravene the *Computer Misuse Act 1990* and with conspiracy to commit false accounting. They are bailed to appear before *Bow Street Magistrates Court* on 24th July.

The court case is expected to be delayed for several months to allow investigators to sift through the enormous volume of hardcopy and over a gigabyte of disk-based material, in a variety of formats, seized at the defendants' homes. Using the conspiracy charges will enable the *Crown's Prosecutor* to demonstrate to the Court the full enormity of this case as all three defendants will face trial together.

New Scotland Yard sources reveal that a number of the victim sites were unaware that they had been targeted; detectives will be contacting all known victims over the next few weeks. While the police are confident they have rounded up all 8LGM's UK members, they know that this group has members in other countries.

Long-standing *VB* readers will know that *New Scotland Yard's Computer Crimes Unit* is also responsible for the collation of evidence regarding computer virus attacks. The unauthorised modification of computer systems is an offence under *Section 3* of the *Computer Misuse Act*; this has been interpreted to cover computer viruses, which by necessity modify programs and/or boot sectors.

The *Computer Crimes Unit* can be contacted by telephone on 071 230 1176 or 1177.

TECHNICAL NOTES

The GP1 Mystery Unravelled

A short updated and amended report on the GP1 virus (first reported by *VB* in June of this year) appears on page 9 of this edition. One of the difficulties in reporting malicious programs which target proprietary software is that specific knowledge of the software's exact operation is not generally available to the researcher.

VB is indebted to *Novell* for its assistance in unravelling some of the mysteries behind GP1. Contrary to our original published report, the GP1 virus does not attempt to gain privileged access on to the network. Instead, it attempts to broadcast passwords to a 'trawler' program resident on a network node. Eric Babcock of *Novell* terms this program 'EARS' due to its presumed ability to 'listen' (or collect) password information. The GP1 virus samples received by *Novell* were not supplied with the associated 'EARS' program so analysis must remain somewhat hypothetical.

In live testing, the GP1 virus replicates in much the same way as the standard Jerusalem virus from which it was derived (see p. 14). It is believed that the GP1 virus was used in testing network security on a specific LAN in Holland although no further information has become available.

The results of various experiments with live computer viruses on *NetWare* are also published in a report on pp. 10-18. The most important conclusion of the report is that network administrators should distinguish clearly between *NetWare* **rights** and **attributes**. Attributes are part of *NetWare's* workstation environment emulation, while rights are *NetWare's* own security and access control system. *Attributes* provide no protection against viruses, while the proper use of *rights* offers substantial protection against virus propagation.

The Invisible Twin

One of the viruses included in this month's list of new arrivals is the Twin-351 virus. It belongs to a small group of companion viruses, which includes AIDS II and TPworm.

Companion viruses have been described before in the *Virus Bulletin* - they are unique in that they do not actually change the files they 'infect'; instead they exploit the fact that DOS executes a COM file before a corresponding EXE file.

The virus creates a new COM file for each EXE file it 'infects', and when the user attempts to run the EXE file, the COM file containing the virus will be executed instead. The virus does whatever it is designed to do, and finishes by loading and executing the EXE file. To avoid detection, all the known companion viruses set the 'hidden' attribute bit.

The Twin-351 virus adds a new twist to this method. It remains resident in memory, and hooks into INT 21H. When the FindFirst function is called, the virus traps the call, thus preventing the FindFirst function (and any subsequent FindNext function) from finding any hidden files. By definition, this makes the virus a stealth virus, as it does not make any apparent changes to any programs, and takes active steps to prevent detection of itself while active.

Most virus scanners use the FindFirst/FindNext functions to locate the files they scan, so they will not find the virus while it is active in memory. However, virus scanners which read the directory on a sector-by-sector basis will encounter no problems in detecting it.

ANSI Bombs and Trojans

Recently a large batch of malicious programs arrived indirectly from one of the larger virus 'exchange' BBSes. In addition to the usual collection of new viruses, it included an 'ANSI bomb generator'. The purpose of this program is to assist in the creation of escape sequences, which could then be incorporated in a text file.

The escape sequences use the key-redefinition ability of ANSI.SYS: if the TYPE command is used to display the file containing such an escape sequence, one or more keys on the keyboard could be redefined. For example the Z key might be redefined as '<ESC>DEL *.*<ENTER>Y<ENTER>', which would delete the files in the current directory if the user pressed the Z key while at the DOS prompt.

Trojan horse writers often use embedded escape sequences intercepted by the ANSI.SYS driver, which is loaded by a command in the CONFIG.SYS file on many PCs. Redefining 'A' as 'X' or 'F' as 'T' may cause confusion, but redefining 'R' as 'DEL *.TXT' (for example) could have more serious consequences. This is easily done. The following sequence

```
<ESC>[082;"<ESC>DEL *.TXT";13p
```

(where <ESC> is the Escape character, hexadecimal 1B), incorporated in a README file is an example of a typical ANSI Trojan. The unsuspecting user uses the TYPE command to display the contents of the file README, and in so doing unwittingly redefines the key 'R'. Each time he presses 'R' thereafter, the keystroke is expanded by ANSI.SYS to 'DEL *.TXT' followed by a carriage return. More devious schemes can be devised. Bulletin Board operators (SysOps) normally search all messages for escape sequences to prevent unsuspecting users downloading this type of Trojan. The easiest way to combat this type of Trojan is to eliminate the statement

```
DEVICE=ANSI.SYS
```

from the CONFIG.SYS file.

This method of key-redefinition is old and well-known and several replacements for ANSI.SYS exist with this feature

disabled. As some of them are smaller and faster than ANSI.SYS, they might be a better choice in most cases. Most applications today do not use ANSI.SYS escape sequences to output to screen but call the BIOS routines directly.

Unfortunately, a new way to abuse ANSI.SYS has now been discovered, which makes it possible to execute a program on a diskette, just by issuing the DIR command. This method could be used to activate a virus, but so far it has not been used for a malicious purpose.

The Useless Virus Simulation Program

The purpose of a virus simulator is, quite naturally, to simulate a virus in some way. A few simulators have an educational purpose and may even be quite entertaining. They simulate some of the effects of viruses, such as playing tunes or producing visual effects such as the falling letters display of the Cascade virus or the bouncing ball display of the Italian virus. The only problem with this type of virus simulator is that it may give the impression that all viruses are harmless - they only produce strange effects on the screen or strange sounds coming from the speaker, which is far from the truth.

Recently a virus simulator with a different purpose turned up. The shareware program (available for US$15.00 by *Darian Rosenthal, Rosenthal Engineering, 3737 Sequioa, San Luis Obispo, CA 93401, USA*) generates a set of other programs (boot sectors, COM and EXE files), which contain bits and pieces from actual viruses, but are harmless in themselves. These viral fragments are obtained from published virus identification strings, including those from the *Virus Bulletin*, from IBM's *VIRSCAN*, and from various other products. The intention of the author is to provide a method for comparing the detection capabilities of virus scanners, which would not require access to live viruses.

There are some fundamental flaws in Rosenthal's approach. The most serious flaw is its inability to judge the performance of any non-signature-based virus scanner, or a scanner which uses a set of signatures to which Rosenthal does not have access. The most secure scanners use proprietary search data and only employ published search strings as supplementary search data. Moreover, different virus scanners often use different but equally valid hexadecimal strings. Even if a scanner did recognise one of the identification strings included in the file, it might not identify the file as being infected - for example because the string was located in an obviously incorrect position in the file.

Rosenthal's virus simulator is of no use whatsoever, and may do more harm than good - for example by resulting in the selection of an inferior virus scanner - simply because its signatures were included in Rosenthal's database. There are immense commercial pressures on software developers to submit their search data for inclusion in such a simulator despite the fact that its conception is completely misguided.

LETTERS

Sir,

I am writing to try to clear up a rather surprising confusion that I have evidently caused. In a reply to a letter in the July 1991 *VB*, the editor reprints a posting of mine to *VIRUS-L* and attributes to me by implication the opinion that, among other things, virus scanners should not scan for viruses not known to be in the wild, or thought to be extinct. I would like to state that this is not my opinion, and that I did not intend to give the impression in my posting (which in fact consists mostly of questions, not of statements or opinions). The fact that the IBM Anti-Virus product scans for at least as many 'research' or 'collector only' viruses should serve as evidence to the contrary. In fact, I doubt that any other anti-virus workers who have taken part in the discussion would agree to the theory of 'selectivity' as the editor states it.

On the other hand, I would like to take this opportunity to outline a view that I would support, which I hope is sufficiently far from the naive 'selectivity' view to avoid confusion. As the anti-virus field moves beyond the butterfly collector stage and into a more mature and responsible era, anti-virus workers will quite naturally move beyond the simple questions of how many viruses they can find, and the details of what a specific virus does. To be of the maximum service to our customers and the community, we have to say more than 'we know of these 400 viruses and only if you buy the product can you be saved; the Snorfler virus, for instance, will erase all your data on alternate Thursdays.' We must also be able to give some idea of which viruses are in fact the most serious threat, which are likely to become threats in the future and what anti-virus measures are likely to be most effective (after all, any *VB* reader knows how to protect a single machine against all the most common viruses; the difficulty now seems to be to figure out how to protect an entire community or organization.)

In order to do accurate threat-estimation, and research into how viruses behave at the organizational or societal level, we need to know new kinds of things about viruses. We need to know what software sharing patterns are like, what causes some viruses to be common and others not, and which viruses are in fact common in the real world today. It is toward the answers to these sorts of questions that our most interesting current work is focused, and it was in an attempt to attack some of these questions that I made my posting to *VIRUS-L*. I think that we in the anti-virus community do need to be selective, but not by simply ignoring viruses that are not in the wild. We need to be selective, instead, about where we concentrate our research, and to be sure that we don't ignore the important large scale questions because we are using all of our resources on just gathering all the viruses we can find.

It's perfectly acceptable, and accepted, for an anti-virus worker today to say to the press 'there are over six hundred viruses in the world', for a software maker to advertise the product primarily on the number of viruses it detects, and for a publication to rate products primarily on that number. In the future, though, I would hope that a responsible researcher would at least add ' although only about 10 percent of them are actual threats', that a responsible software maker would at least say 'including the 10 percent known to be in active circulation', and that a responsible publication would give the reader some idea of how a product performed against the most important subset of their complete test-set. Similarly, it would be very nice if collectors exchanging viruses with trusted peers would also exchange anything they know about the history or current status of the viruses involved, and not simply binary samples. I trust that as the industry continues to mature, all these good things will happen.

One statement in the editor's reply with which I would definitely disagree: he states that no functioning virus can be classified as a 'non-threat'. That is not the case: anyone with much experience in the anti-virus field can say with confidence that the MGTU virus, while it is technically a functioning virus, will never become more widespread than a non-trivial 'arf-arf' style Trojan horse would; the virus is just too slow-spreading and obvious. To claim that we can always tell which viruses are the dangerous ones would of course be foolish; but to claim that we have no idea and that all viruses must be treated the same way because they are all threats, would be equally inadvisable and I am sure that the editor did not mean to make that claim.

Thanks for the chance to clear this up!

David M. Chess
IBM T J Watson Research Center, New York

[Apologies to David Chess for any misinterpretation arising from the editor's alcoholic rantings about 'selectivity'. There's much food for thought in this letter which may well stimulate further debate. *Ed.*]

Sir,

Having just read with interest the tutorial concerning boot sector viruses and recovery, in July 1991 edition of *VB*, I thought you or your readers may be interested to read the following corrections/amplifications.

The article was informative and suitably clear, however I did notice that it suggested that *FDISK* could be used to remove a Master Boot Sector virus. Unfortunately I have found this not to be the case. *FDISK* (or at least all of the many versions I have used), is a Partition Table editor. It allows you to create and delete logical partitions. If it detects what it thinks is a valid boot sector program in residence it will not replace it, but it will just allow you to edit the Partition Table element.

So it is possible to delete all of the DOS partitions with *FDISK* and (depending on the version of *FDISK*), lose all of the data held within the partitions. However the virus would still be in place and when you recreated the same or different partitions and started to boot from the drive, the virus would be active again!

To force *FDISK* to rewrite the boot record, you need to use a disk editor and remove the signature 55AA from the last two bytes of the boot sector. Then when you run *FDISK* it will allow you to recreate the partition table and rewrite a valid boot sector program over the top of any virus code.

Also on the subject of removing virus code from DOS boot sectors, a FORMAT is suggested. While this will indeed remove the virus it will obviously remove any data as well. An equally effective and less hasty solution is the humble SYS command which removes the virus leaving data intact.

While in the writing mood, may I change the subject and raise another point for discussion. What follows are my personal opinions:

Firstly, I must say that I look forward to receiving *VB* and enjoy reading it immensely. However, I am tiring now of some of your reviewers' aggressive stance against *any* new approaches to the virus problem. [1]

Whenever I see a review by Keith Jackson I know exactly what to expect. He will complain about lack of an index [2], then he will tell me that the product confused him and finally he will launch into an outright slanging match against these filthy people who have dared to offer a product that could possibly stop a virus. [3]

I have no doubt that some products are less than perfect, but surely it is in all computer users interests to encourage any attempts to stem the tide of virus activity. There will always be bad reviews but *all* of Keith's reviews are bad! [4] Do you just give him the dross to keep him quiet or is he trying to become the Nina Myskow† of the virus world? [5]

As a comparison I offer the review by Fridrik Skulason in the same issue. This *is* a review, he offers the pros and cons so the reader can make their own mind up and keeps his personal opinions to a minimum. [6]

In Keith Jackson's review of the *Knoxcard* (with which I have absolutely no connection) he has the audacity to write that the card manufacturer is lying when they say that a single virus could not disable all the various *Knoxcards*. I am not suggesting that they are right or wrong, but if they suggest that this is correct they should be given the benefit of the doubt.

The final straw came when he actually suggested that the *Virus Bulletin* or one of its reviewers should set out to crack the cards security. When I read this I was appalled! [7]

Is this magazine for the anti-virus community or hackers and the virus writing community? I believed and hoped the former was the case. [8]

The *Virus Bulletin* has earned it's place in the anti-virus community, letting reviewers express personal views with which they have little to back up, not only opens that reviewer but also the *Virus Bulletin* itself open to ridicule and possibly even legal repercussions.

Yours sincerely,

Kevin Powis
Visionsoft Ltd.

[The tutorial article '*Fixed Disk Boot Sectors and Post-Attack Recovery' (VB*, July 1991, pp. 5-9) never stated that a Master Boot Sector virus could be removed using *FDISK*. It said instead that *FDISK* could be used to edit the Partition Table, which is correct. *Ed.*]

Dr. Jackson comments...

[1] I strongly resent the charge that I complain about new methods of combating viruses. I complain about methods that don't *work.*

[2] Manuals without indexes are not helpful to the *user*. It is a sad comment on the standard of most manuals that this must continually be mentioned.

[3] If I have ever resorted to a slanging match then I humbly apologise. I notice that Mr. Powis offers no example of such conduct.

[4] As for all my reviews being bad, I let the record stand for itself. I've recently written favourable reviews of *Dr. Solomon's Anti-Virus Toolkit* and *VISCAN*. On the other hand when products do not perform, I feel obliged to point this out - I am answerable to *VB's* general readership and not anti-virus product manufacturers such as Mr. Powis.

[5] Nina Myskow! If only I was paid as much as her!!!

[6] As for just offering the pros and cons of a product, consider the last two reviews by myself published in *VB*. They were a scanner program that detected <10% of the virus test-set and a hardware product that kept locking up my computer. Am I really supposed to overlook such shortcomings?

[7] This was a joke written in total frustration, of course I did not intend that *VB* should actually do this!

[8] To claim that *VB* has done anything else than help combat viruses is just plain silly.

Finally, I plead guilty to the charge of criticising deeply flawed products. Developers should make sure that a product *works* before marketing it and not complain when reviewers discover gaping holes in their oft hyped-up claims. *KJ.*

† Nina Myskow is a British TV critic noted for her scathing reviews.

5th July 1991

Sir,

We certainly do not claim that *Knoxcard* cannot be reverse engineered as your reviewer claims (*VB*, July 1991, p. 39). Any competent systems programmer should be able to achieve this. What should be kept in mind, however is that once a piece of software is unassembled, writing a program to override it depends on knowing the contents of specific locations/addresses within the unassembled code. And what the *Knoxcard* User's Guide explains is that these locations are not common across all *Knoxcards* and they are mixed to get an infinite number of combinations, thereby preventing anybody from writing a common piece of code to override the *Knoxcard* virus checks.

Yours sincerely,

SURESH. K.
Knoxware, India

Sir,

Thank you for taking the time to evaluate *Trend Micro Devices PC-cillin Virus Immune System*. Apparently, the outdated version of *PC-cillin* (V2.95) that Mr. Hamilton reviewed had a compatibility problem with QEMM and 386^{MAX}. This problem prevented Mr. Hamilton from installing and fully testing *PC-cillin* against the *Virus Bulletin's* viruses, thereby affecting his results.

On behalf of *Trend's* defence, I would like to clarify two very important points. First, without installing the TSR intelligent viral traps, *PC-cillin* would be unable to detect all viruses, as Mr. Hamilton pointed out. *PC-cillin* would be limited to detecting only those viruses contained in the *Quarantine* or pre-installation scan. This point emphasizes the importance of *PC-cillin's* traps which search for symptoms of a newly discovered virus, rather than relying only on a scan pattern bank of known viruses which will always be (despite the increasing number of annoying updates) an ineffective and soon to be obsolete, method of virus protection.

Second, although it is possible to save boot sector data on a diskette or by using *The Norton Utilities*, the only way to achieve automatic, virus-free, boot sector recovery is by using *PC-cillin's* isolated hardware immunizer.

In order to present your readers with an equitable review of *PC-cillin*, I feel that a fair representation of both sides should be addressed. Please consider publishing the above comments regarding *PC-cillin*.

Thank you,

Steve Chang
Trend Micro Devices

Mark Hamilton comments:

Mr. Chang's opening remarks concerning third party memory managers disguise the fact that *PC-cillin* had obviously not been properly tested prior to its release.

More importantly, why did his company, at the end of May, supply *VB* with a version for review that had already been superseded? Is this ill-fated version still being supplied to his customers?

I would suggest that using a dongle to store essential boot sector information is considerably less secure than storing it as a file on an off-line diskette. We are already witnessing the emergence of viruses that target specific high-profile anti-virus products and *Trend's* dongle could well be within the virus-writers' sights. If *Trend's* software can read from, write to and interrogate its own dongle, then so can a virus - how secure is your boot sector now?

Mr. Chang does make one very important point worthy of elaboration. Given the spiralling number of viruses, it will soon cease to be practical to provide every end-user with virus-specific detection software, as it will impact too heavily on the PC's resources. This point has been raised many times by *Virus Bulletin* - the search for practicable and secure generic defences continues unabated.

Finally, I was horrified to see an advertisement for *PC-cillin* in a recent issue of *PC User* magazine which declared that this product 'Kills all known viruses. Dead'. It doesn't.

Referring to the product's dongle as a 'Hardware Immunizer' is daft. The dongle is simply a new use for an existing, outmoded and unnecessary copy protection device which does not magically immunise a PC against viruses. What utter tosh! The advertisement further claims *PC-cillin* 'is unique as it uses both software and hardware components'. It isn't, what about *Thunderbyte* from *Novix International* or, indeed, the ill-fated *Knoxcard*? (*VB*, July 1991, pp.38-40)

The *Advertising Standards Authority* should investigate *Trend's* UK distributor's advertising claims which, in *VB's* opinion, contravene at least two out of the three *ASA* tenets - *legal*, *honest* and *true*.

M.H

LETTERS & FAXES

We welcome letters and faxes. These should be sent to the *VB* office no later than the fifteenth of the month. The ideal *VB* letter is short, concise, witty, interesting and controversial and arrives in ASCII readable text on an IBM PC compatible diskette of any density. Hard copy can be sent by fax:

Fax 0235 559935
Fax International +44 235 559935

IBM PC VIRUSES (UPDATE)

Updates and amendments to the *Virus Bulletin Table of Known IBM PC Viruses* as of July 21st 1991. Hexadecimal search patterns can be used to detect the presence of the virus with the aid of a disk utility program, or preferably a dedicated virus scanner.

200 - CN: When an infected program is run, this 200 byte virus infects all COM files in the root directory of drive C:

```
200             33D2 B800 42CD 218B CEB4 40CD 212E 8B0E
```

337 - CR: A small, simple virus which does nothing but replicate.

```
377             5FBF 0001 578B CC2B CEF3 A433 F633 FF33
```

Arab, 834 - CR: Awaiting analysis.

```
Arab            3D00 4B75 368B EC8B 7600 8B7E 028C C98E
```

Delirium - CER: Yet another Murphy variant from Italy. 1778 bytes long and detected by the HIV pattern.

Captain Trips - CER: A Jerusalem variant,1808/1813 bytes long, with modifications intended to invalidate various scanner strings.

```
Captain Trips   03F7 2E8B 8D11 00CD 218C C804 1000 8ED0
```

Dewdz - CN: This 601 byte virus adds itself in front of the files it infects. Displays the text “Kewl Dewdz!” on screen.

```
Dewdz           434B 7409 B44F CD21 72BA 4B75 F7B4 2FCD
```

Fingers 08/15 - CER: A 1322 byte virus which is awaiting analysis.

```
Fingers 08/15   AE26 803D 0075 F847 4747 8BD7 1E2E 8C16
```

Jerusalem-1361 - CER: A stripped-down version of the Jerusalem virus, with all unnecessary code (including the trigger) removed.

```
Jerusalem-1361  218C C805 1000 8ED0 50B8 2F00 50CB FC06
```

Jerusalem-Clipper - CER: 1408/1413 byte Jerusalem variant. It generally infects EXE files, (no COM files were infected in testing).

```
Jeru-Clipper    2E8E 1612 002E 8B26 1000 2EFF 2E14 0058
```

Kemerovo-B - CN: Similar to the original Kemerovo virus, but appears to have been assembled with a different assembler.

```
Kemerovo-B      0400 8BF8 B904 00A4 E2FD 8BFA 2BDA 81EB
```

Lazy - CR: A primitive 720 byte virus, which always occupies the same area in memory and may cause system crashes if a large program is run. The major effect of the virus is a slowdown of the infected computer.

```
Lazy            1E84 0026 A186 008E C026 8B07 BB90 5029
```

Leech - CR: A 1024 byte virus which uses self-modifying encryption, making the extraction of a search pattern difficult.

```
Leech           FA1E 078B EC8B E681 C4E4 038C
```

Leprosy-D - CN: A 370 byte overwriting virus, derived fom one of the earlier variants. Infected programs must be deleted.

```
Leprosy-D       B43B CD21 4683 FE03 7CE6 EB00 5EC3 8B16
```

Milan Overwriting - CN: A group of primitive, overwriting viruses from Italy. Two variants are known - BadGuy, which is 265 bytes long and does nothing but replicate, and Exterminator which is 451 bytes long. When Exterminator activates it overwrites the beginning of the hard disk, destroying the FAT and root directory of drive C:

```
Exterminator    02EB E2B4 2ACD 213C 0174 03EB 2F90 C606
BadGuy          02EB D9B4 2ACD 213C 0174 11EB 1D90 071F
```

Mosquito - ER: A 1024 byte virus which is awaiting analysis.

```
Mosquito        5650 BE49 002E 8A24 2E32 261E 002E 8824
```

Mule - CER: A 4112/4117 byte encrypted variant of Jerusalem. First reported in Australia. Detected by the Jerusalem-1 pattern.

Shadowbyte - CN: A 723 byte virus which is awaiting analysis.

```
Shadowbyte      8B54 0183 C203 B442 CD21 89F2 83C2 03B9
```

Stardot-600 EN: This virus may be related to the September 18th virus. It overwrites the beginning of logical drives when it triggers.

```
Stardot-600     32F6 B908 0033 DB51 B901 00D1 C250 CD26
```

Twin-351 - CR: A companion type virus which attempts to hide from detection while memory-resident.

```
Twin-351        8C4C 048C 4C08 8C4C 0CB8 004B 8D16 0F01
```

Vienna-733 - CN: An encrypted variant of Vienna. It activates if an infected program is run on the second day of the month and produces a high-pitched sound.

```
Vienna-733      89D6 81EE F201 89F7 B956 01FC ACFE C0AA
```

Virdem 824 - CN: A new version of the Virdem family. The following pattern can be found in all the Virdem variants.

```
Virdem-family   83C3 1C26 C707 205C 431E 8CC0 8ED8 8BD3
```

NOVELL UPDATE

Eric Babcock
Novell Inc., Provo, Utah, USA

Novell's Analysis of the GP1 Virus

[As indicated in Jim Bates' article on the GP1 virus (*VB*, June 1991, pp. 5-7), further investigation into the functioning of this virus continues. *VB* is grateful to Eric Babcock, *Novell's* software security manager at the company's US head offices, for supplying the following updated and amended report on GP1 which clarifies its *NetWare*-specific functioning.]

In June of this year UK virus researcher Jim Bates provided *Novell* with a copy of the original GP1 code and a thorough analysis and disassembly. A GP1 sample from *McAfee Associates* confirms that we are talking about the same code as everyone else. The code is a Jerusalem virus derivative with the trigger and file deletion code (and a few other odds and ends) replaced by code designed to provide someone in an organisation with other peoples' password information; hence the name 'Get Password One' and the *NetWare*-specific code.

The *NetWare*-specific code in the GP1 virus:

- ➤ tests for the presence of a *NetWare* shell at the workstation.
- ➤ checks for a specific form of login request by the workstation. This form of login request does not use encrypted passwords.
- ➤ Broadcasts the login information in the login request via IPX if the proper login request occurs. The socket number (2A9FH) in the broadcast packet is a value not likely to be used by any other program. The code to perform this task is non-functional in the samples, but could easily be corrected.

Socket numbers in broadcast packets control which machines on the LAN will accept the broadcast. *NetWare* file-servers accept requests addressed to socket 0451H. A workstation's IPX device driver monitors broadcasts, accepting packets addressed to sockets opened by workstation applications. Workstations discard broadcasts with unopened socket numbers.

GP1 is designed for use on a specific network where a separate non-viral application is operating on a workstation. This non-viral application would collect the broadcasts from the GP1 virus in other workstations and store the login information from these workstations. *Figure 1.* illustrates the situation, with the workstation running the non-viral application labelled 'EARS'. The owner of the 'EARS' workstation is also the creator of GP1. The 'EARS' part of the GP1 virus application was not provided with our GP1 samples.

NetWare supports the login function call checked for by GP1 when 'allow unencrypted passwords' is on. *NetWare* 2.xx and 3.xx *NetWare* login utilities do not use this function.

GP1 is not known to have spread beyond its original location. In the absence of an 'EARS' workstation, this virus is limited in the damage it can cause. Possible damage may include spreading to other files, using up memory in workstations and slighly increasing network traffic.

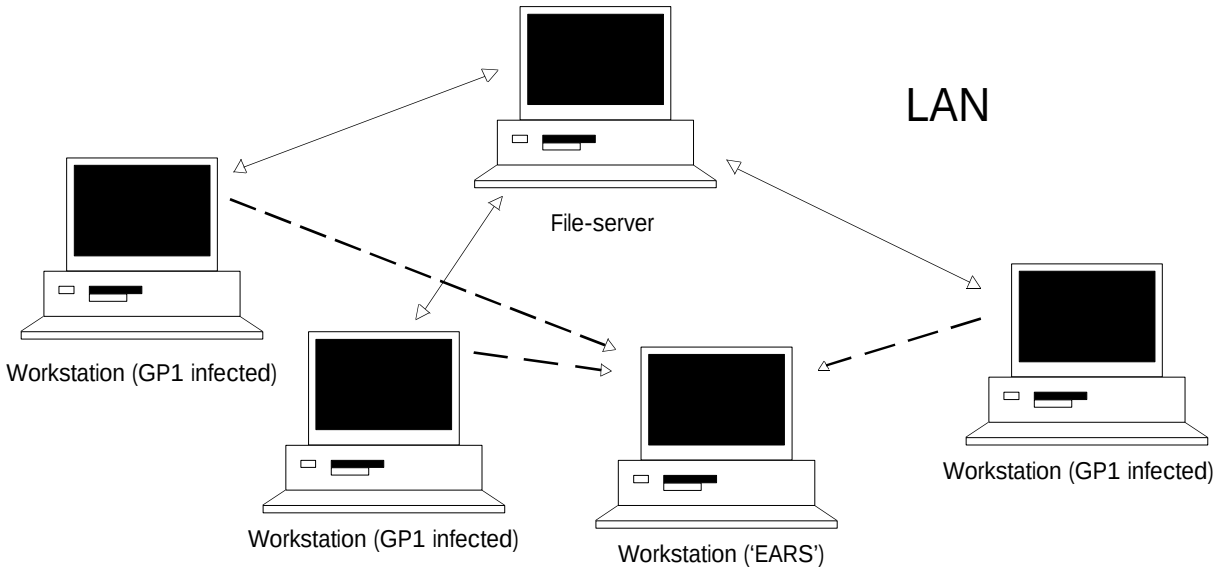


Figure 1. Login information exchange with a GP1 infected LAN

NOVELL EXPERIMENTS

Dr. Jan Hruska / Richard Jacobs

Virus Propagation and NetWare Security

Computer viruses spread through interchange of executables between computers. On Personal Computers (PCs) this interchange is much more frequent than on minicomputers and mainframes. This report will concentrate on viruses on PCs.

The interchange of executables on non-networked PCs is almost exclusively done by floppy disks and is, as a consequence, relatively slow and physically controllable. PC networks allow high speed interchange and sharing of data and executables. This interchange is also much more difficult to control in practice, with hundreds of simultaneous users.

The danger of a large scale virus attack in a non-networked organisation is comparatively small. The attack will be limited to a few PCs before it is spotted and disk interchange is prohibited. The possibility of a large scale virus attack on a network is much greater and the chances of containment smaller, if proper network security features are not used.

This report concentrates on *Novell NetWare* 3.11 and is a result of a theoretical and practical study of virus behaviour under *NetWare* 3.11 and *NetWare* 2.12. Although practical anti-virus measures described are specific to *NetWare* 3.11, much of it applies also to other network operating systems such as *IBM LAN Manager*. It is assumed that the network will be running on a dedicated file-server.

VIRUS TYPES AND REPLICATION MECHANISMS

A virus is a deliberately written computer program which usually consists of two parts: self-replicating code and the 'payload', which produces side-effects. In a typical PC virus, the replicating code may have between 400 and 2000 bytes, while the size of the payload will depend on the side-effects. Typically this is a few hundred bytes.

The side-effects of a virus are limited only by the imagination of the virus author and can range from annoyance to serious vandalism.

Virus Types by Point of Attack

Viruses can be divided into four categories according to the executable items which they infect: parasitic viruses, boot sector viruses, multi-partite viruses and companion viruses.

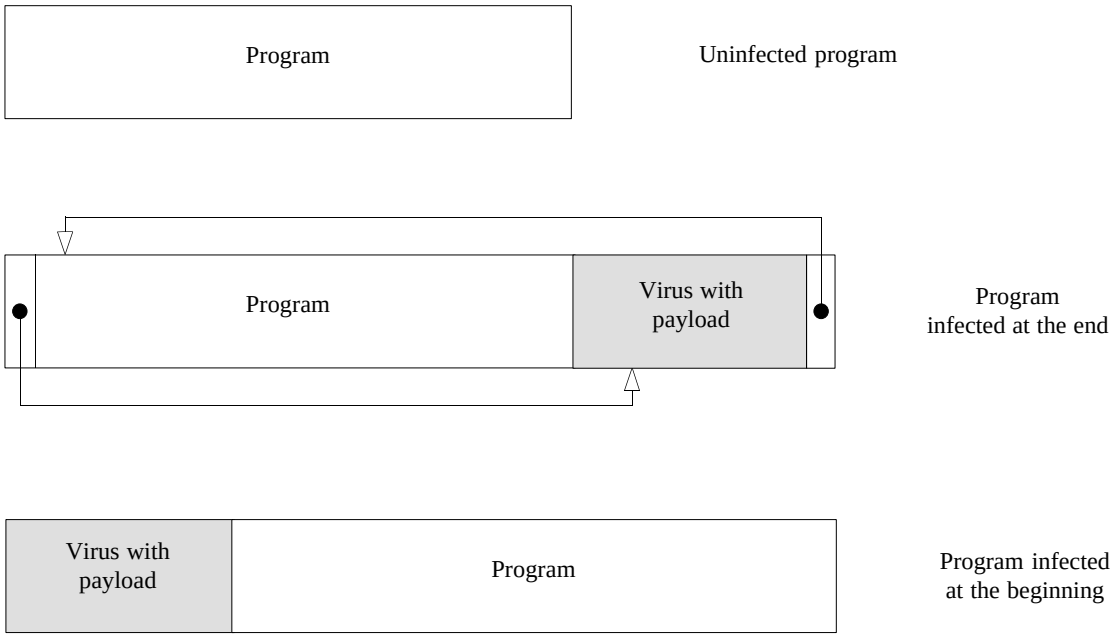


Figure 1. Program infection with a parasitic virus

Parasitic Viruses

Parasitic viruses modify the contents of COM and/or EXE files. They usually insert themselves at the end, or at the beginning of the file, leaving the bulk of the program intact. The initial jump instruction in the program is modified, but program functionality is usually preserved, although there are several viruses which overwrite the first few hundred bytes of the program rendering it unusable. When an infected program runs, the virus code is executed first. The virus then returns control to the original program, which executes normally. The extra execution time due to the virus is normally not perceptible to the user. (See *Figure 1*.)

Most parasitic viruses, such as Cascade, spread when another (uninfected) program is loaded and executed. Such a virus, being memory-resident, first inspects the program for infection. If it is not infected, the virus will infect it. If it is already infected, further infection is not necessary (although some viruses such as Jerusalem do reinfect *ad infinitum*). Other viruses do not install themselves in memory, but spread by finding the first uninfected program on disk and infecting it. An example is the Vienna virus.

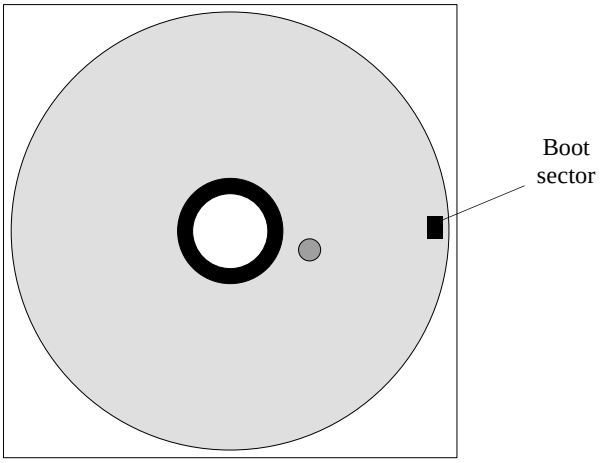


Figure 2. Uninfected disk

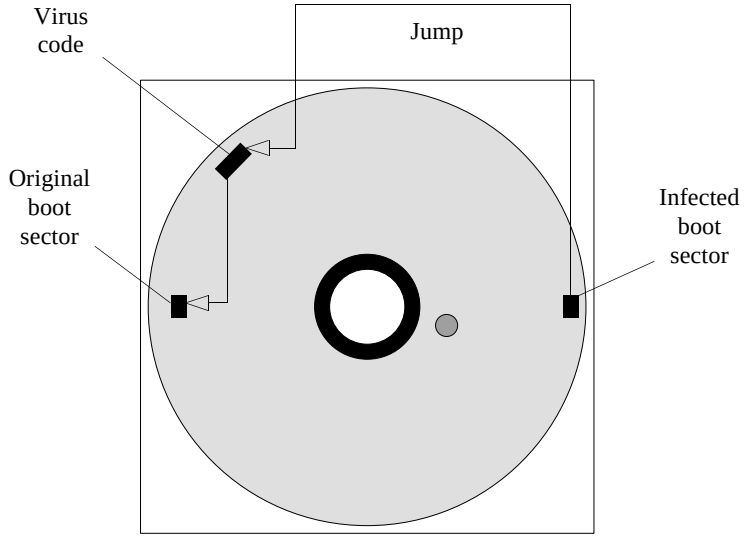


Figure 3. Disk after boot sector virus infection

Boot Sector Viruses

Boot sector viruses modify the contents of either the Master Boot Sector or the DOS Boot Sector, depending on the virus and type of disk, usually replacing the legitimate contents with their own version.

The original version of the boot sector is normally stored somewhere else on the disk, so that on bootstrapping, the virus version will be executed first. (See *Figures 2* and *3*.) This normally loads the remainder of the virus code into memory, followed by the execution of the original version of the boot sector. From then on, the virus remains memory-resident until the computer is switched off.

A boot sector virus is thus able to monitor and interfere with the action of the operating system from the very moment it is loaded into memory.

Examples of boot sector viruses include Brain (floppy disk boot sector only), Italian (floppy disk and hard disk DOS Boot Sector) and New Zealand (floppy disk DOS Boot Sector and hard disk Master Boot Sector).

Multi-Partite Viruses

A comparatively recent development has been the emergence of viruses which exhibit the infective characteristics of both boot sector viruses and parasitic viruses. For example, the Flip virus (see *VB*, Sept. 1990, pp 18-21) infects executable files (COM and EXE) as well as the Master Boot Sector of hard and floppy disks.

Companion Viruses

Companion viruses exploit the DOS property that given two programs with the same name but different extensions, the operating system will execute a COM file in preference to an EXE file. A companion virus creates a COM file for every EXE file it 'infects'. The COM file is usually marked 'hidden' and contains the virus code, which also executes the EXE file.

Companion viruses do not spread widely in practice, since the DOS COPY command does not copy 'hidden' files.

VIRUS BEHAVIOUR AFTER INFECTION OF THE PC

Memory-Resident Viruses

Memory-resident viruses install themselves into memory as Terminate and Stay Resident (TSR) process when an infected program is executed. They will normally intercept one or more interrupts and infect other executables when certain conditions are fulfilled (e.g. when the user attempts to execute an application (Cascade) or when the user accesses a drive (Brain).

Switching the PC off will clear the virus from memory (though not from disk); warm bootstrapping with Ctrl-Alt-Del may not, as some viruses such as Yale and Joshi intercept and survive this process.

Non-Memory-Resident Viruses

Non-memory-resident viruses are active only when an infected application is executed. They execute their code completely at that stage and do not remain in memory. Other executables are generally infected only when an infected program is executed (e.g. Vienna or Datacrime).

The infectiousness of non-memory-resident viruses is just as high, if not higher, than that of memory-resident viruses. They are also more difficult to spot, since they do not change the interrupt table or the amount of available memory, and their infectious behaviour can be more unpredictable.

PATHOLOGY OF A VIRUS INFECTION ON NETWARE

Due to *NetWare's* excellent emulation of physical DOS disks, many DOS viruses in existence today are able to attack *NetWare* drives.

The main difference between *NetWare* and local workstation drives is that *NetWare* does not allow individual sector addressing either through the normal DOS interrupts 25H and 26H or the BIOS interrupt 13H.

This excludes the possibility of pure boot sector viruses infecting the network, but does not, of course, exclude parasitic, multi-partite and companion viruses, all of which can spread freely on a badly protected network.

Virus Entry Into the Network

A virus will usually enter a network via the user workstation. In a typical scenario, the user infects his workstation by executing an infected application (parasitic or multi-partite) or by booting from an infected disk (multi-partite viruses). The virus becomes memory-resident and will typically try to infect any application which is run, or any drive which is accessed. NET3 and IPX, which are normally kept on the workstation, may already be memory-resident at this stage.

On accessing the network the user executes LOGIN.EXE, stored on the file-server, which opens access to the allotted file areas on the file-server. If LOGIN.EXE itself, or any other executables, are unprotected (see page 18), they will become infected. Any user executing an infected application will have his workstation infected, which, in turn, will spread the infection.

On a typical active network, infection can spread onto most workstations within minutes. An infected LOGIN.EXE, or any program executed by the system login script, causes user workstations to become infected whenever any user logs into the network.

JERUSALEM INFECTION ON NETWARE 2.12

The above scenario has been demonstrated by intentionally infecting a workstation with the Jerusalem virus and then executing LOGIN on the file-server running *NetWare* 2.12.

LOGIN.EXE was purposefully left protected only with Read-Only (R/O) attributes by logging in as a supervisor. Jerusalem (like most parasitic viruses) sets the R/O attribute to Read/Write (R/W), infects the file and resets the attribute to R/O. After LOGIN.EXE has been infected, any workstation logging into the network will become infected. Any EXE or COM file residing on the file-server will likewise be infected whenever executed by the supervisor.

A Jerusalem infection is easy to spot because of virus side-effects, which include system slow-down and the appearance of a black 'window' on the screen some 30 minutes after infection. Infected EXE files keep growing by 1808 bytes every time they are executed from a workstation infected with the virus; this does not happen with COM files.

NETWARE 3.11 SECURITY MECHANISMS

NetWare 3.11 provides four different aspects of network security: the login procedure, trustee rights, inherited rights mask and file/directory attributes.

- ➤ The login procedure requires all users to identify themselves by a username and a password.
- ➤ Trustee rights are granted to each user by means of trustee assignments and allow each user various actions such as reading from files, writing to files, creating files etc.

- ➤ The inherited rights mask of a directory determines the effective rights of that directory (read, write, open, close, delete, search) which are set separately and can be used to limit access to certain directories such as those containing executables. Trustee assignments override the directory effective rights.
- ➤ File/directory attributes (read-only, read-write, share) can be set separately.

Even if a user's PC becomes infected, the infection cannot spread to the file-server, if the security features are properly implemented. This security does break down if the network supervisor's PC becomes infected. Care should be taken when setting network security features, as the appropriate features may not be enabled by default.

NETWARE 3.11 PRACTICAL EXPERIMENTS

An experimental network consisting of a dedicated file-server (on a Compaq 486/25, 310 MByte hard disk, 4M RAM) and a workstation (Amstrad PC-ECD, 20 MByte hard disk, 640K RAM) was set up with default security parameters.

Parasitic Viruses

It was decided to investigate *NetWare* 3.11 resistance to attack with different levels of protection. A workstation not logged in was infected with Jerusalem (memory-resident, parasitic virus). IPX was executed (and infected) and NET3 was executed (and infected). From then on, any COM or EXE file did **not** become infected when run; this applied to files held on floppy, hard or network drives. The interaction between the virus and NET3 appeared to prevent the virus from infecting other executables.

If the sequence is reversed, i.e. if a clean workstation is loaded with IPX and NET3 and then infected, the following error message is produced:

```
Network Error on Server SERVER: Error receiving
from network
Abort, Retry?
```

This error arises because Jerusalem uses INT 21H function E0H to check whether it is memory-resident. This function is also used by the *NetWare* print command. When the virus issues this function call, *NetWare* intercepts it and tries to send a print command leading to unpredictable results.

The same trial was repeated with Cascade and Vacsina, and in both cases the viruses lost the ability to infect immediately after infecting NET3.COM. Unlike Jerusalem, Cascade and Vacsina did not crash the workstation if loaded after IPX.COM and NET3.COM.

The same trial was then undertaken with the 4K virus. The virus did infect IPX and NET3, did not crash the workstation and proceeded to be infectious in its normal way on floppy and hard disks, but not on the file-server.

The test was repeated with the Eddie-2 virus. A clean workstation was logged into the network and an infected application executed from drive A:. This virus proved infective on all drives, including the file-server.

We then tested the infectiousness of Eddie-2 with various *NetWare* 3.11 file attribute settings. Eddie-2 is a virus with limited stealth capability. It intercepts the DIR *find-first* and *find-next* calls and displays the original file lengths. In order to establish whether a file is infected or not, a secure bootstrap must be performed.

DEFAULT NETWARE 3.11 SECURITY

By default the users have full access rights to their home directory (created at the time of user creation) and no write-rights to any subdirectories containing executables.

The Eddie-2 virus could infect files in the user's own directory, irrespective of the setting of file read-only attributes, but could not infect any other files on the server.

Rights Set To Read-Only

The virus could not infect files to which the user did not have 'effective rights' to write, irrespective of whether this right was denied at a directory or file level, or from the 'Inherited Rights' mask.

File Attributes Set To Read-Only

The virus could infect files which had their file attributes set to read-only. This attribute is the same R/O attribute used by DOS and set by Eddie-2 (and most other parasitic viruses) to R/W before infection and reset back to R/O after infection.

File Attributes Set To Execute-Only

NetWare 3.11 allows file attributes to be set to execute-only and such files cannot be read even by the supervisor. An Eddie-2 infected workstation was used to execute an execute-only file as well as a file marked read-only. The workstation was rebooted. Looking at the file DIR entries, the execute-only file was not infected while the read-only file was.

Running Under Supervisor Mode

Supervisors have all rights to all directories and files. A clean workstation was used to log into the network as a supervisor which was then infected with Eddie-2.

The virus was able to infect all files on the file-server, except those marked as execute-only.

Boot Sector Viruses

Although boot sector viruses have no means of infecting a network drive (since *NetWare* does not allow individual sector addressing), an experiment was nevertheless undertaken.

A workstation was infected with the New Zealand virus, which infects the Master Boot Sector on hard disks and the

boot sector on floppy disks. The network was accessed (LOGIN followed by running of various applications, followed by LOGOUT). The workstation was cleared of infection and the network connection was re-established. The workstation hard disk and the workstation memory were examined for infection, and, as expected, nothing was found.

Multi-Partite Viruses

A clean workstation was used to log into the file-server. The workstation was infected with the Flip multi-partite virus. Files on the local fixed disk could be infected as usual, but when files on the file-server were executed, DOS returned the message:

```
EXEC Error
```

In general a multi-partite virus infects files on a network drive in the same way as a parasitic virus, but in addition the virus infects the boot sectors of disks attached to the workstation.

NETWARE SPECIFIC VIRUSES

There are three cases of viruses reported to have been written specifically to circumvent *NetWare* security.

First 'Novell Virus'

In February 1990 there appeared an (unconfirmed) report of a '*Novell*' virus which supposedly destroyed the *Novell*-specific file allocation table.

The virus was said to be capable of penetrating a file-server from a workstation even if it was not logged on to the network. It was suggested that this might be possible by altering the NET$DOS.SYS program by using C libraries released by *Novell*.

Novell has not encountered this virus, nor has the company received any reports of it. Neither *Sophos* nor *Virus Bulletin* have had any further reports about this 'virus' apart from the Editorial in *Virus Bulletin* in February 1990.

Dr. Jon David

In July 1990 New York consultant Dr. Jon David released a report about a virus which he observed propagating on a *Novell* LAN. Dr. David said that the virus, a Jerusalem mutation, bypassed *NetWare* file-server write-protection and also deleted write-protected files on the server.

After a heated exchange in the press and the *Virus-L* Bulletin Board between Dr. David and *Novell* (at one point *Novell* was threatening to sue Dr. David), *Novell* confirmed that the virus was Jerusalem, that it did propagate on unprotected networks, but denied the allegation that it bypassed *NetWare* security.

The most disturbing fact was that Dr. David refused to disassemble the virus himself or release his sample to a responsible organisation for analysis. He preferred to observe the virus effects, rather than analyse the virus structure.

The universal conclusion seems to be that the virus was a standard copy of Jerusalem with no specific ability to subvert *NetWare* security.

NetWare Virus From The Netherlands

In April 1991 *Virus Bulletin* received a virus (GP1) from Holland which contained instructions to subvert *NetWare* security. Interestingly enough, the virus was received in source-code form. It is reported to have been developed in Leiden (Holland) as a result of an unofficial challenge by a state organisation employee to a student.

GP1 Virus Structure

The virus is based on the Jerusalem virus, with *NetWare*-specific instructions added to the disassembled version of Jerusalem. The virus is memory-resident but contains no stealth features. The *Novell* network handler is accessed via a FAR JMP call instead of a FAR CALL; analysis indicates that if the FAR JMP instruction is changed into the FAR CALL instruction, the virus could become fully functional.

The virus is not infective unless it is run on a *NetWare* workstation. It intercepts four different INT 21H services, of which the most interesting is the *NetWare*-specific service E3H. This is checked to see whether the sub-function requesting the service is a user LOGIN procedure. If it is, the LOGIN is executed under the control of the virus and the return code is examined. If the LOGIN is successful, the virus sends a copy of the original login request block to the socket number 2A9FH. We suspect that this is a broadcast message (for more information see page 9).

Practical Trials On NetWare 2.11

The virus was assembled after changing the FAR JMP to a FAR CALL instruction. An experimental network consisting of a dedicated file-server (on a Compaq 386s, 80 Mbyte hard disk) and a workstation (Amstrad PC-ECD, 20 Mbyte hard disk) was set up with default security paramaters.

The virus replicated in the same way as Jerusalem (when *NetWare* was present), but no other effects were observed.

The background of this virus continues to be investigated and it appears that the copy obtained was an unfinished version.

Practical Trials On NetWare 3.11

An experimental network consisting of a dedicated file-server (on a Compaq 486/25, 310 Mbyte hard disk, 4 Mbyte RAM) and a workstation (Amstrad PC-ECD, 20 Mbyte hard disk, 640 Kbyte RAM) was set up with default security parameters.

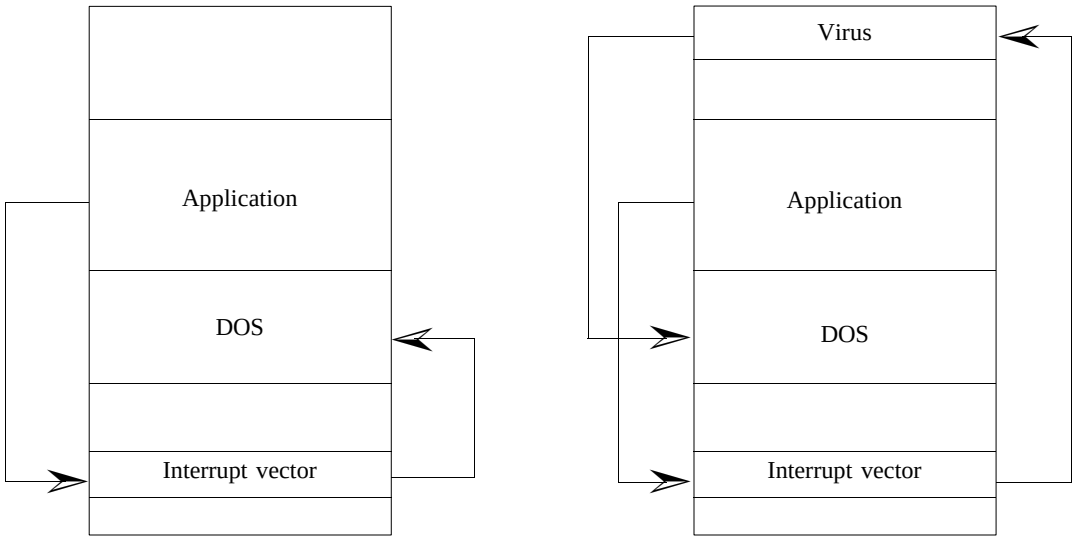


Figure 3 - Interrupt routing before and after infection

The GP1 virus was tried under *NetWare* 3.11, where it replicated without problems, unlike the standard Jerusalem virus which refuses to replicate under the same circumstances. After becoming memory-resident the virus infects other files, extending them by 1546 bytes.

There were no other visible side-effects.

HIDING MECHANISMS

Viruses often place obstacles in the path of anyone trying to find them or eradicate them. Two mechanisms are commonly used: interrupt interception and encryption of the virus program itself.

Interrupt interception in particular has special implications on any network, due to the difficulty in establishing a 'clean', virus-free work environment.

Interrupt Interception

The virus redirects the interrupt vectors in such a way that operating system service calls are redirected to the virus code first. For example, the virus can examine every request made to DOS for reading disk information. If the sectors requested are those used by the virus, their contents are falsified before further processing of the request. (See *Figure 3*.)

This is the tactic used by the Brain virus, which intercepts any call to read the disk bootstrap sector and substitutes the original contents in place of the virus-infected actual contents.

Encryption

Certain viruses encrypt their own contents in order to foil attempts to find the virus by disassembly or by searching for a characteristic pattern. Since the encrypted part of the virus can be made different for each infected program, a simple pattern check can not discover its presence; the only search possibility is on that portion of code which performs the decryption. Likewise, disassembling such a virus using standard tools is likely to be a convoluted process. The virus must first decrypt its own contents before executing. This is the tactic used by the Cascade virus, which performs rudimentary self-encryption using a very simple exclusive-OR operation. The decryption routine of this virus remains static thus enabling the extraction of a search pattern. (See *Figure 4.*) However, some viruses such as 1260 modify the decryption routine itself, so that it is impossible to extract a conventional hexadecimal search pattern. Although encryption complicates the development of detection software it does not impose any specific additional burdens on network security.

IMPLICATIONS OF HIDING MECHANISMS ON NETWARE 3.11

The main problem of dealing with stealth viruses on any network is the difficulty in establishing a positively 'clean' work environment from which the cleanup can be attempted (see p.16, 'Secure Accessing of *NetWare* 3.11').

Interrupt interception represents a particular problem when dealing with an infected network. Viruses such as 4K hide

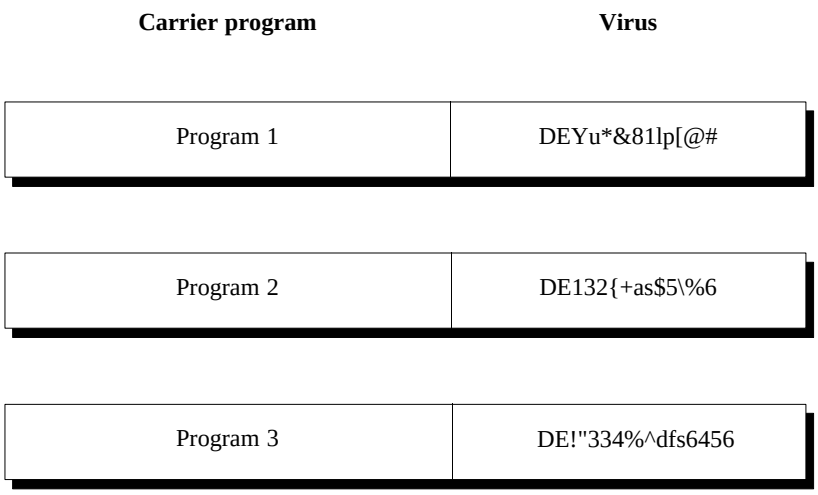


Figure 4 - Three programs infected with an identical encrypted virus

their presence by intercepting about fifteen different interrupt services, including file-open and file-close. The virus disinfects each file on opening it and re-infects it on closing, which means that any software checking for the virus' presence will not discover it if the virus is active in memory at the time of checking.

PRACTICAL ANTI-VIRUS MEASURES FOR NETWARE 3.11 NETWORK ADMINISTRATORS

Diskless Workstations

Diskless workstations are PCs in their own right, sometimes equipped with hard disks, but without any floppy disks. The security reasons for equipping users with diskless workstations include the hope that if the user has no means of introducing floppy disks into the PC, he will also have no opportunity to introduce a virus. This 'no-floppies no-virus' reasoning holds only up to a certain extent. It is quite true that diskless workstations will help prevent accidental introductions of viruses onto the network. However, the prevention of malicious introduction of viruses is not guaranteed, as the virus code can still be input through the keyboard using the DOS COPY command. The technique is described in Burger's *Computer Viruses - A High Tech Disease*. Likewise, diskless workstations can still have modem connections over which software can be downloaded from BBSs.

The major disadvantage of diskless workstations is that the transfer of data by users is made much more difficult. Moreover, users have no means of taking backups locally at workstations. The decision to use diskless workstations is a major one. Associated implications for the efficiency of the organisation should be carefully assessed.

Remote Bootstrap ROMs

Most network cards can be fitted with a special Read Only Memory (ROM) chip which maps into the PC memory space and when executed on boot-up, reads the operating system and other associated files from the file-server, instead of from the local disk.

There are several advantages in using remote bootstrap ROMs. Firstly, the technique eliminates the danger from boot sector virus infection.

Secondly, any updates to the operating system used are made much easier, since they can be done on the file-server. The use of remote bootstrap ROMs is recommended for bootstrapping both diskless workstations and individual PCs connected to the network.

Enhanced Access Control

NetWare 3.11 provides very good access control features and utilities for the administration of users. A number of access control packages are available which front-end *NetWare* 3.11, providing even more sophisticated access control features and, perhaps, easier administration of users.

ANTI-VIRUS SOFTWARE

Two types of detection software can be used: virus-specific and virus-non-specific. The authors recommend standalone (application) software; memory-resident anti-virus software is not recommended.

Regardless of which type of software is used, proper procedures must be followed to ensure that the machine running anti-virus software is clean, i.e. free of any virus active in memory. If this is not the case, stealth viruses can use hiding techniques to prevent the software from discovering them (see 'Secure Accessing of *NetWare* 3.11' below).

Virus-Specific Software

A virus-scanning program relies on the knowledge of known virus 'patterns'. When a new virus appears in the wild, it is analysed, and a characteristic pattern of some 16-32 bytes recorded. The virus-scanning program scans all executables on a disk, including the operating system and the boot sector(s), and compares their contents with the known virus patterns.

This type of software can only discover viruses that it 'knows' about and as such has to be updated continually with new patterns, as new viruses appear. This is the main problem with scanning software.

The use of virus-specific software on networks is recommended since the problems associated with updating the master copy are minimal: one copy can be held on the file-server and updated easily. The checking process can be performed overnight, minimising the network workload.

It is vitally important that the workstation used to initiate the scanning is booted from a clean write-protected system disk. Viruses such as Dark Avenger infect files as they are opened; if such a virus were resident in memory as scanning proceeded, it could infect every file stored locally on the workstation and, more significantly, on the file-server itself.

Checksumming Software

Checksumming software relies on the calculation of a checksum of any executable on the system followed by periodic recalculation in order to verify that the checksum has not changed. If a virus attacks an executable, it will usually change at least one bit of the executable, which will result in a completely different checksum (providing a strong checksumming algorithm is used). The exception is a special class of viruses known as companion viruses which do not change files (see page 12). However, well implemented checksumming software will report modifications such as the bogus hidden COM files which these viruses create.

This type of software is reactive rather than proactive, in that a virus attack will be detected after it happens. Checksumming software also relies on the fact that the executables are clean (i.e. virus-free) before initial checksumming is applied. This can be ensured by using virus-specific scanning software to check the system for the presence of known viruses.

The checksumming approach is the only known method which will detect all viruses, present and future, with absolute certainty. The method of performing the checksumming process (the checksumming algorithm) is very important. Three general approaches are possible: Simple checksums, Cyclic Redundancy Checks (CRCs) and cryptographic checksums. The results of the checksumming algorithm must not be easily reproducible (lest a virus should do this on infection, preventing its detection).

It is recommended that checksumming software is used on *NetWare* 3.11 in a fashion similar to the virus-specific software. The main problem is deciding which areas of the file-server should be fingerprinted and checked regularly. On *NetWare* 3.11 it is recommended that all executables in the \PUBLIC, \SYSTEM and \LOGIN subdirectories are fingerprinted. In addition, each system will have subdirectories containing applications software; these should be fingerprinted as well. Checking of the fingerprints is best done from a separate, securely booted workstation. This should be done before performing backups as well as at a specific time during the night on a daily basis.

TWO IDs FOR SYSTEM ADMINISTRATORS

One of the weak points in any multi-user computer system is that one or more users must be given high privileges necessary for system administration. Unfortunately, **these privileges are also assigned to a virus whenever it is in control of a workstation logged in as a network supervisor.**

One way of reducing the danger from virus penetration via this route is to reduce the time that network supervisors are logged in as network supervisors. They should ideally have two user IDs, one with all privileges and the other with limited privileges. The use of the former should be limited to system administration functions and supervisors should be extremely cautious of using it if a virus infection is suspected.

SECURE ACCESSING OF NETWARE 3.11

With the advent of stealth viruses, it is most important to guarantee a clean, virus-free environment before running anti-virus software on a network (Note that the following procedure presumes that the remote bootstrap ROM is **not** in use.)

To access *NetWare* 3.11 securely, prepare a system disk containing the DOS system files, COMMAND.COM and the following *NetWare* 3.11 files:

- IPX.COM
- NET3.EXE
- LOGIN.EXE
- MAP.EXE

Write-protect the floppy disk.

To access the network, switch the PC off, boot from the floppy disk and then run IPX first, followed by NET3. Run LOGIN from the floppy disk.

Check that the system login script or the user login script does **not** contain the command

```
COMSPEC= ...
```

since this causes a potentially infected COMMAND.COM to be loaded from the network when needed. If that statement is present, issue the command

```
COMSPEC=A:\COMMAND.COM
```

If the login script contains any programs which are automatically run from the network whenever a user logs in, the script will have to be changed so that no software residing on the server is used. If the network requires a particular package to be used during the login process, a positively clean copy of that package should be added to the floppy disk and the login script on the server should be modified so that the package is executed from the floppy disk.

TIGHTENING NETWARE 3.11 SECURITY

NetWare 3.11 allows the setting of file attributes to execute-only. This prevents file modification or reading by any user, including the system supervisor - the only thing that he can do (apart from executing them) is to delete them.

Setting the execute-only attributes has mixed blessings. On the one hand it prevents the modification of executables, but on the other hand it makes them unreadable (and unverifiable) by anti-virus software. We recommend that this attribute is **not** used and that instead write-rights are removed from directories containing executable files.

SUMMARY

NetWare 3.11 Administration

- ➤ Set *NetWare* 3.11 directory and user rights correctly.
- ➤ Do not rely on default *NetWare* 3.11 attribute settings.
- ➤ Do not use *NetWare* 3.11 execute-only attributes unless absolutely necessary.
- ➤ Use secure bootstrap procedure before running anti-virus software.

NetWare 3.11 Virus Infections

- ➤ *NetWare* 3.11 seems to cause more memory-resident viruses to malfunction than *NetWare* 2.12.
- ➤ Some memory-resident parasitic viruses interact with IPX and NET3 losing the ability to infect. Some memory-resident parasitic viruses crash the workstation if IPX and NET3 are already loaded when the virus-infected application is run.
- ➤ Most parasitic viruses will infect *NetWare* 3.11 files protected with a Read-Only attribute.
- ➤ Parasitic viruses do not infect *NetWare* 3.11 files when the user's effective rights do not include 'write' rights. Supervisor has 'write' rights to all directories.
- ➤ Parasitic viruses do not infect *NetWare* 3.11 files with execute-only attributes set, regardless of the user.
- ➤ Boot sector viruses do not infect *NetWare* 3.11 drives.
- ➤ Multi-partite viruses will infect unprotected *NetWare* 3.11 executables.
- ➤ Parasitic and Multi-partite viruses will infect executables regardless of protection levels (execute-only files excepted) if the user is logged in as a supervisor.

Other Considerations

- ➤ Consider using diskless workstations
- ➤ Use remote bootstrap ROMs in workstations

It is very important clearly to distinguish between *NetWare* **rights** and **attributes**. *Attributes* are part of *NetWare's* workstation environment emulation, while *rights* are *NetWare's* own security and access control system. *Attributes* provide **no** protection against viruses, while the proper use of *rights* offers substantial protection against network virus infection and propagation.

Bibliography and References

F. Skulason, 4K, A New Level Of Sophistication, *Virus Bulletin*, May 1990
J. Bates, A *Novell*-Specific Virus, *Virus Bulletin*, June 1991
R. Burger, *Computer Viruses, a High-Tech Disease*, Abacus, 1988
Dr. F. Cohen, *A Short Course on Computer Viruses*, ASP Press, 1991
Editorial, *Virus Bulletin*, February 1990
Editorial, *Virus Bulletin*, December 1990
R. Glath, Virus Propagation on *Novell*, *Virus Bulletin*, December 1990
Dr. H. J. Highland, *Computer Virus Handbook*, Elsevier, 1990
Prof. L. J. Hoffman, *Rogue Programs: Viruses, Worms and Trojan Horses*, Van Nostrand, 1990
Dr. J. Hruska, *Computer Viruses and Anti-Virus Warfare*, Ellis Horwood, 1990

VIRUS ANALYSES

Jim Bates

2100 and 'Cracker Jack' the Plagiarist

The recent 'explosion' of new virus variants has increased the workload of researchers to an almost unbearable extent and this is thought to be an inevitable result of the opening of virus 'exchange' Bulletin Boards all over the world.

Computer viruses are a fascinating subject for study and quite naturally therefore, they can be expected to arouse general curiousity and interest. However, the 'research' disguise that such BBS systems adopt should be seen to be just that - a disguise! Genuine virus researchers have long since established their own communications links around the globe and have no need to exchange virus code with public access Bulletin Boards.

The suggestion that anyone can become a 'researcher' by downloading a virus and attempting to take it apart is pure eyewash - akin to being given heroin/guns/explosives so that one can 'experiment'! Certainly the anti-virus community has urgent need of genuine and dedicated researchers, but it should be understood that the true researcher would never consider even modifying a virus let alone writing a new one. Yes, there are undoubtedly 'researchers' who have written viruses, but their irresponsibility and lack of integrity in an extremely difficult field will disqualify them from ever attaining the respect of their contemporaries. No public access Bulletin Board should ever have viruses (either as object code or source) available for download and legislation is well overdue to stop this malicious trade.

Plagiarism

It has always been accepted that copying and modifying an existing virus is much easier than writing a new one from scratch and the increasing availability of virus code in both binary and source forms is giving the plagiarists the opportunity to copy some of the more sophisticated viruses as vehicles for their own twisted ideas.

A case in point has come to light during research into one of the Dark Avenger 'targeting' viruses, 2100. Pattern recognition scanners indicated similarities between this and several newly received viruses of Italian origin. Further research indicated that the Dark Avenger viruses were being admired, copied and modified by an Italian virus writer calling himself 'Cracker Jack'. The new range of viruses are variously named HIV, Migram and Smack (a.k.a. Patricia). They include sections from Dark Avenger 2100 and another Dark Avenger copy known as Murphy. The new code added by Cracker Jack displays a laughable ignorance of basic programming techniques but the combination of code sections simply confirms the extreme dangers of virus exchange trading.

2100

Let us first examine the original virus, known as Dark Avenger 2100 after its infective length - this is known to be at large in the UK and has caused problems at several sites. 2100 is a 'targeting' virus; it deliberately sets out to circumvent known anti-virus software written (in this case) by Vesselin Bontchev in Bulgaria.

When a file infected with 2100 is first executed, the virus checks for the existence of highly specific sections of code. The first of these checks examines the address of various interrupt handling routines to see whether the virus is already resident. Then a check is made of both RAM and ROM, looking for specific indications of resident anti-virus software.

This checking of RAM has been encountered before, but the ROM examination routines are much less common and demonstrate how a determined hacker can easily avoid the sort of protection provided by the various add-on boards which are now becoming available. When these checks are completed, various flags and entry point addresses are collected within the virus code and the virus then installs itself into high memory and hooks intercept routines into various system services. The list of functions and services subverted in this way is long and bears examination.

Interrupt Services

INT 13H - Hard Disk BIOS access
INT 21H - DOS Function services
INT 24H - Critical Error handler
INT 27H - TSR handler

Function Calls (via INT 21H)

11H	- FCB FIND FIRST
12H	- FCB FIND NEXT
25H	- GET VECTOR
35H	- SET VECTOR
31H	- TERMINATE STAY RESIDENT
3CH	- CREATE FILE
3DH	- OPEN FILE
3EH	- HANDLE CLOSE
43H	- CHANGE ATTRIBUTES
56H	- RENAME FILE
4B00H	- LOAD AND EXECUTE
4B01H	- LOAD, NOT EXECUTE
4EH	- HANDLE FIND FIRST
4FH	- HANDLE FIND NEXT
5BH	- CREATE FILE

Figure 1. System services subverted by the 2100 virus. Developers of memory-resident virus monitors beware!

This list (*Figure 1.*) gives some idea of just how comprehensively this virus attempts to monitors system services.

Stealth Features and 62 Seconds Stamp

All the familiar stealth capabilities are present including the subtraction of virus code from reported file lengths for infected files. However, the virus uses the very old method of marking its own infection by setting the time field to 62 seconds. This signature produces some interesting results since some software deliberately sets its time field in this fashion (in an ill-informed attempt to prevent infection) and is therefore reported as being 2100 bytes shorter than it really is when 2100 is memory-resident. Under the right circumstances, this causes incorrect loading of files marked in this way with consequent corruption and malfunctioning of the machine.

Some software vendors still insist on marking their products in this way (presumably under the misconception that this will give them protection against viruses); they should realise that such a practice simply makes their software more likely to fail when certain stealth viruses are active in memory.

Interrupt Interception

The 2100 virus prevents attempts to change certain system vectors (using 'legal' DOS procedures) but 'fakes' the results and thereafter erroneously reports the effects, so that simple virus detection software will be unaware of the changes. Similarly, programs attempting to Terminate and Stay Resident are hooked into the system in a way that the virus can still remain hidden and in control.

These techniques present enormous obstacles to the development of resident anti-virus monitoring programs; these processes must be clearly appreciated before any such monitoring software is designed.

Trigger Routine

There is a selective trigger routine which only comes into operation if the virus locates the Bontchev software. This routine has not been copied in any of the other viruses under discussion here - Migram, Smack or Murphy - and it would be irresponsible to publish exact details of what this is or how it works. I suspect that the plagiarists did not recognise it for what it was and therefore left it out of their own creations. However, I can report that during tests, the results of the trigger routine varied considerably from machine to machine and usually resulted in a general failure to the point at which a power-down reboot was necessary. Actual corruption of data stored on disk did not occur during testing and seems unlikely.

There are two highly specific areas in which this virus causes concern: one is in the ROM search routine which appears targeted initially at the machine BIOS but may also identify certain anti-virus add-on boards. The other is in a section of code which addresses and utilises the services of a device driver to access the fixed disk and modify the boot sector. This modification is **not** part of the infection process but seems to remove a particular protection mechanism employed by the anti-virus software or firmware being targeted.

Both of these routines prove the assertion made long ago that there is no such thing as a 100 percent defence against viruses (except perhaps by switching your PC off permanently!), regardless of whether hardware or software is used. However, the point is that 2100 is one of the more sophisticated viruses and contains stealth routines which cause difficulties for simple virus defence programs.

Summary - 2100 Virus

The virus infects COM and EXE files (including COMMAND.COM) but ignores files with the SYSTEM attribute set. It is an appending, stealth, targeting virus with an infective length of 2100 bytes. The code is not encrypted. The trigger routine is only effective if Vesselin Bontchev's anti-virus software is found. A reliable search pattern is:

```
D3E8 408C D103 C18C D949 8EC1 BF02 00BA
```

The Murphy Viruses

The Murphy viruses contain text suggesting they were written by 'Lubo and Ian' who are reported by Vesselin Bontchev as being Lubomir Mateev Mateev and Iani Lubomirov Brankov - both from Bulgaria.

There are at least three known variants of the original Murphy virus and although these are awaiting a full dissection, preliminary disassemblies have been completed in which large sections of code similar to that used by Dark Avenger have been found. This is yet another indication of the unoriginality and poor technical capabilities of the writers. The infection routine has been identified reliably and differs from that used in the Dark Avenger viruses. It is this routine which has been copied by Cracker Jack in his attempts to produce his own viruses.

The Migram and Smack Viruses

With the exception of the trigger routines, these two viruses are identical their operational code. It appears that Migram came first since it is comprised of almost 'straight' code.

A second version (Migram-2) is identical save for two NOP instructions placed at strategic points (where no assembler would place them) and possibly designed as an experiment in disrupting pattern recognition searching. This hypothesis is supported when the Smack virus is examined and found to contain an inordinately large number of NOP instructions inserted seemingly at random throughout most parts of the code (excepting the portions copied from 2100 and Murphy).

Both Migram and Smack contain slavishly copied sections of the 2100 code which examines the ROM. However, in this case the writer displays almost total ignorance of exactly what the code accomplishes and does not make proper use of the information collected. Similarly, the EXE file infection routine from the Murphy viruses has also been copied exactly without obvious awareness of its operation.

Like 2100 and Murphy, Migram and Smack are also resident viruses and install their own INT 13H, INT 24H and INT 21H handlers. The code is not encrypted during infection and no attempt at stealth is made once the code becomes resident and operative. The Trigger routines might best be described as 'unusual', but more of this later.

Installation

In this instance both viruses make use of an 'are you there?' call to the system by placing the value 4B4DH in the AX register and issuing an INT 21H function call. This will return with the carry flag set if the virus is **not** resident, or cleared if it is and the virus will install itself or exit to the host accordingly. The next routine is that copied from 2100 which examines ROM (and EPROM) areas for a suitable entry point into the disk BIOS. The actual code fragments which the viruses look for are:

```
cmp dl,80h      or      test dl,80h
jnc ??                  jnz ??
int 40h                 int 40h
```

if either of these is found in ROM, it is used as an access point to the disk BIOS.

The code then continues through a series of calculations designed to install it into high memory without recourse to the normal TSR function calls. Finally, the host program is repaired and processing is passed to it.

Operation

The INT 21H intercept routine in these viruses only checks for LOAD and EXECUTE (4BH) and FILE OPEN (3DH and 6CH) function calls (only calls to open for READ ONLY access are intercepted). Obviously there is a recognition/answer routine for the 'are you there call' but all of the other functions are intercepted by the same routine.

In Migram, the interception routine collects the name of the file being processed and examines it for a .ZIP or .EXE extension. If neither is found, processing is allowed to return to normal DOS operation. When either a .ZIP or a .EXE file is located, it is opened and examined for the presence of the 'MZ' header. If the file does not contain this header it must be a ZIP file and a separate routine is called which searches the current directory for the first ZIP file and deletes it. This deletion occurs regardless of the system date or time setting.

When a file is found which contains the 'MZ' header (the rarer alternative 'ZM' is not checked for), a check is made of the system date and if the weekday indicator shows Saturday, then a trigger routine is called. On days other than Saturday, an attempt is made to infect the file before processing is returned to the caller.

With Smack, the interception routine is similar, but the conditions of 'acceptance' are different, as are the resulting actions. In this virus, COM and EXE files are identified by the last two letters of the file extension ('OM' and 'XE'). In the case of 'OM' files, a further routine tests for a filename ending in 'ND' and thereby excludes COMMAND.COM and similarly matching files from any further interference. With 'XE' files the situation is a little more involved and checks are made for names ending with 'AN', 'HA' and 'HK'. Attempting to execute any file which matches these criteria (e.g. SCAN.EXE or VIRUSCHK.EXE) while the virus is resident will result in the system attempting to reboot through INT 19H. This is an obvious attempt to avoid detection - the *SCAN* program from *McAfee Associates* being the most widely used virus scanner in the world.

All other files are examined for the presence of the 'MZ' header (again the 'ZM' possibility is not considered). If the header is not found, processing passes to a routine which checks the system date see whether it is a Saturday - if it is the file is deleted, otherwise the file is infected. If the 'MZ' header is found, a different trigger routine is executed but if the system date is anything other than a Friday, the file is infected. On Fridays, a series of infantile messages is displayed as follows:

```
Is today Friday?
```

The virus then waits for the user to press the 'Y' key. If the user answers 'Y' then the virus displays:

```
Sorry but on Friday I wish not work!!
```

and exits back to DOS.

If the users answers anything but 'Y' to the original question, the virus displays:

```
You are untruthful!! For punishment I format your
HD Fat!!
```

and then proceeds to execute a similar routine to that found within Migram which is apparently intended to format the first few tracks of the hard disk. In both viruses this routine appears to have been written by a complete novice. The routine attempts to format the first five tracks of drive C: but fails for several reasons.

We are not in the business of training virus writers or in giving them marks out of ten (although if Cracker Jack were a plumber, he would have drowned years ago). The bugs in the code will therefore not be reported individually.

Suffice it to say that the display of relatively sophisticated code alongside a plainly infantile mess gives these viruses a strange appearance when disassembled.

Also within the code for the Smack virus are plain text messages which are not displayed during virus operations. These are as follows:

```
This virus was written in Italy by Cracker Jack
1991 IVRL All rights reserved, please don't crack
this virus!!

Special message to Patricia Hoffman: I love
you!!!!!!!! SmackSmack!!

Can you give me your telephone number??? Ciao
bellissima!
```

Seasoned *VB* readers will know that Patricia Hoffman maintains a regularly updated listing of known IBM PC viruses which is widely distributed as a shareware text file. (*Technical Editor's note*: 'Cracker Jack' has expressed dissatisfaction with researchers renaming the 'Patricia' virus to 'Smack' - one of his viruses contains the string 'Smack Virus ....What a horrible name!!!!!!!!!!!!!!!!!!!' More ominously, the same virus contains the message 'Compliments to the Dark Avenger for the nice viruses...'.)

"If 'Cracker Jack' were a plumber he would have drowned years ago..."

SUMMARY

Migram Virus (two versions)

These infect only EXE files and are simple appending viruses with infective lengths of 1219 and 1221 bytes. The operational code is identical in the two versions and is not encrypted during infection. Any ZIP files opened for read only when the system date indicates a Saturday, will be deleted.

An alternative trigger routine attempts a low level format of the first five tracks of drive C: but fails through incorrect coding. These viruses may be recognised by the Murphy(2) and HIV recognition patterns published in the July 1991 edition of *VB*.

Smack Virus (two versions)

These vary only in their infective length and are simple appending viruses with infective lengths of 1825 and 1841 bytes. A reliable hexadecimal search pattern was published in the July 1991 edition of *VB*.

Conclusions

The fact that an inexperienced 'pimply' has copied code (albeit without knowing exactly how it works) from known viruses into his own 'creations' is nothing new. The fact that such a virus was available to him in the first place is of more concern and even though his feeble attempts have not produced the effect that he desired, it is of paramount importance that he (Cracker Jack) and his 'mentor' (Dark Avenger) be stopped by whatever means are available.

Some time ago, I observed that one of the major advantages that anti-virus researchers had over the virus writers was the collaboration that had been achieved across the world. This advantage is rapidly being eroded since the advent of the virus 'exchange' Bulletin Boards, and as the analyses of the above viruses show, an increasing degree of plagiarism is occurring.

The Murphy viruses were written in Bulgaria and their authors' close proximity to Dark Avenger (maybe they know each other personally) probably explains how the 'collaboration' came about.

It is possible that the obvious plagiarism of 2100 and Murphy within Migram and Smack may not have occurred as a result of virus exchange through a Bulletin Board, but the fact remains that it probably did happen that way.

A lone voice in the UK has recently defended the existence of these boards on the dubious grounds that proscribing them would be an infringement of 'human liberty'. This argument calls into question the possible motives behind such a defence but what utter nonsense! Can it be called an infringement of liberty that poisons, weapons, certain chemicals, explosives and similar dangerous items are not publicly available? Similarly, public access to a range of viruses (especially commented source code) represents a danger that must be prevented. When calling these boards, the general offer of a one-for-one exchange is a positive inducement to callers to write or modify viruses in order to use them as an 'invitation' into the inner sanctum.

Two measures by which this activity might be stopped come immediately to mind - if some system of licensing bona-fide researchers were implemented, unlicensed possession of virus source code or collections of virus samples could be made a criminal offence. Alternatively, intentional transmission of virus code across the public telephone network could be criminalised in a way that would allow the authorities to close down the offending boards immediately. In the United Kingdom, the deliberate and unauthorised insertion of a virus into a computer system is a criminal offence under *Section 3* of the *Computer Misuse Act 1990*. However, the possession of virus code and making malicious programs available for download is not illegal under the terms of this Act. The transmission of virus code via public telephone networks may contravene telecommunications laws in different countries but this remains a legal grey area.

PRODUCT REVIEW 1

Mark Hamilton

ProScan - A Commercial Scanner for the Non-Technical End-User

McAfee Associates of Santa Clara, California, is well known for its *SCAN*, *NETSCAN*, *CLEAN* and *VSHIELD* anti-virus products which are marketed as shareware. However, McAfee also produce a shrink-wrapped anti-virus product called *ProScan* which is a melange of the first three of these programs.

It must be emphasised that *ProScan* is designed for use by non-technically minded end-users rather than 'techies'. It thus comes as no surprise that *ProScan* comes complete with fancy screen displays and bounce-bar menus - the typical interface of products designed for the mass market.

Presentation and Contents

The version of *ProScan* submitted for review consisted of a single, write-protected 360 Kbyte 5.25-inch diskette. Fancy packaging and printed documentation were noticeable only by their absence - shades of shareware. On the diskette were three files: PRO-SCAN.EXE, PRO-SCAN.DOC and PRO-INFO.TXT. The .DOC file (just 4 Kbytes in size) turned out to contain amendments to printed documentation - I rechecked the mailer, but no, definitely no printed documentation there. As things turned out, printed documentation proved unnecessary as ProScan is easy to learn and use.

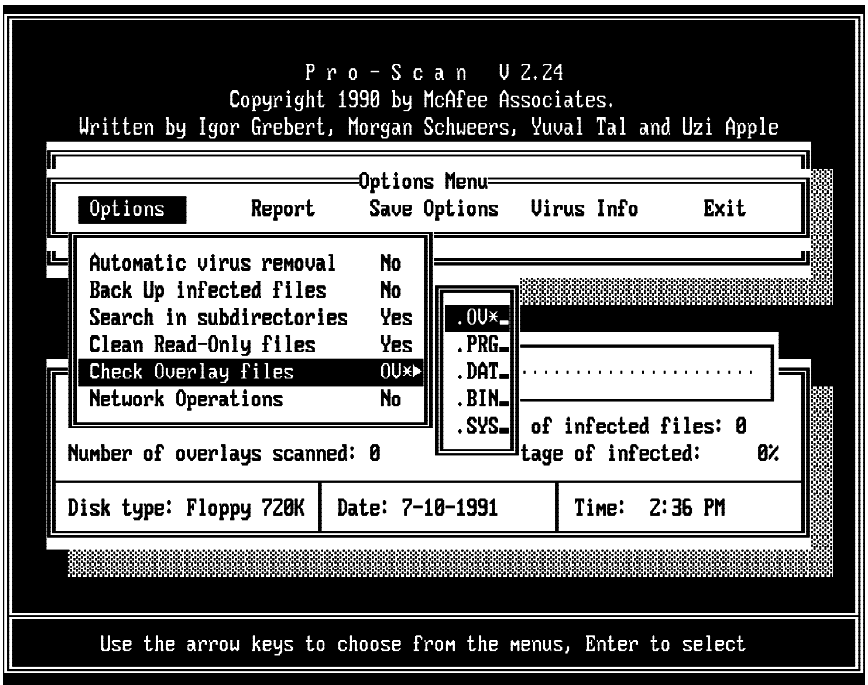


Figure 1. *ProScan* checks .COM and .EXE files by default but can be configured to scan other file extensions using the Options Menu. Note the automatic virus removal option; tests showed this to be an unreliable feature of this program.

The .TXT file contains details of the viruses *ProScan* claims to detect, with each virus' infective characteristics, damage characteristics and infective lengths. It is very similar to VIRLIST.TXT which McAfee distributes with its shareware offerings. This file is read in when the program loads to provide details of the various viruses.

Running ProScan

Installing this product is simplicity itself - you don't need to; it runs straight 'out of the box'. Upon entering the command 'PRO-SCAN', you are greeted by a lurid sign-on screen which is replaced by the initial scanning display as soon as you press a key. You are prompted to enter a search directory, which defaults to the root directory of the drive from which you called the program.

At the bottom of the screen is the prompt 'Press F10 for Options Menu'. You have to remember that key code, because the prompt for it appears only once per invocation of the program. The area of screen upon which it appears is used to display other messages of assistance.

Pressing F10 brings up the options menu upon which there are five principal choices: Options, Report, Save Options, Virus Info and Exit. The Options sub-menu (see Figure 1.) provides the ability to change the operational characteristics of the program. For example you can enable the Network option, so that files can be checked across LANs; you can also toggle the automatic virus removal option and specify what file types constitute 'overlays'. The product always checks files which have the extensions .COM and .EXE and defaults to checking other files with

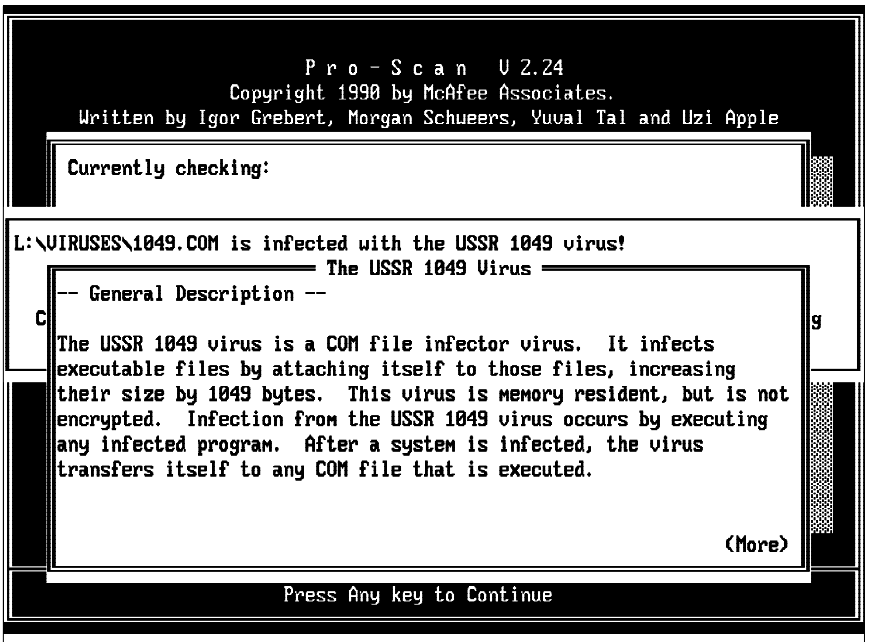


Figure 2. In contrast to most scanning programs, *ProScan*, upon detecting a virus infection, provides cursory information about the offending culprit.

.OV?, .PRG, .DAT, .BIN and .SYS extensions. (See *Figure 1*.) These latter extensions can be toggled on and off. Additional extensions can be included by the user. Wildcard characters ('*' and '?') are accepted. You can also delete unwanted extensions using the Delete key.

The Report sub-menu sets up the report type allowing 'none', 'detailed' (details of all files checked) or 'short' (details of infections found) and the file or device name that is to receive the report.

Unlike other McAfee products, you can obtain information about each of the viruses that *ProScan* claims to detect. When *ProScan* starts up, it reads the contents of PRO-INFO.TXT and uses this to provide cursory descriptions of the various viruses.

You can access information about any of the viruses *ProScan* knows about from within the Virus Info choice in the Options menu. (See *Figure 2*.)

Accuracy Rating

For details of the virus test-suite and testing protocols employed, see *VB* April 1991, p.8 and June 1991, p.34.

ProScan neither checks its own integrity nor performs any memory checks. The shareware programs produced by *McAfee Associates* incorporate basic integrity checking methods (see *VB*, May 1991, p.11) to prevent unauthorised modification once they are released into general circulation. Since *ProScan* is circulated by more secure means, this precaution was presumably considered unnecessary.

ProScan detected 270 of the 365 parasitic infections and six of the eight boot sector infections. These results are partly due to the fact that the version submitted for review (2.24) was created on 21st May 1991. Also, given the international makeup of the *VB* virus test suite, it is possible that some of the strains used have yet to be analysed by the program's authors.

'Disinfection' Capability

Each time an infection is discovered, a menu-box appears with four options: 'Continue checking', 'Remove virus', 'Info on virus' and 'Stop checking'.

The removal option produced some interesting results. When instructed to remove an infection of the Amstrad virus, *ProScan* reported that the virus had been successfully removed and that the file size had been reduced from 384 to 64226 bytes When *ProScan* had completed its scan, I checked to see the exact state of this 'disinfected' file. Sure enough, the Amstrad-infected file had genuinely grown from 384 bytes to 64226! [*This failure was caused by ProScan misidentifying the virus as one of Amstrad's 847 byte variants. Tech. Ed.*] Moreover, a file infected with the Advent virus continued to contain this infection after the file was reported to have been successfully 'disinfected' by *ProScan*.

Virus disinfection is an inexact art which requires an intimate knowledge and understanding of each virus and the ability to identify it exactly and in all instances. Unfortunately, *ProScan* is unable to disinfect virus infections reliably and this removal option should only be used with extreme caution.

As a warning to end-users of virus 'disinfection' software it should be pointed out that *ProScan* is by no means unique in failing in this way - anomalous 'freak' results have been recorded time and time again with so called 'disinfection' programs (unreliable disinfection software will be the subject of an article currently in preparation for *VB*). The secure way to recover from a parasitic virus infection is to overwrite infected files, delete them with the DOS DEL command, and restore from trusted write-protected master software.

When *ProScan* detects a virus, the user can access information about it; this is essentially the same option as the 'Virus info' choice in the Options menu, except that information is restricted to the virus discovered.

In Conclusion

This product would be enhanced if the user could augment its detection capabilities with additional search patterns, such as those published by *Virus Bulletin*. *IBM*, *Bates Associates* and *S&S* (to name but a few) provide this facility - why can't *McAfee Associates*?

ProScan is easy to configure, easy to use and eminently suitable for non-technical end-users. However, its detection capabilities are low in comparison to the current market leaders (which include McAfee's *SCAN*) and its disinfection capabilities are of dubious value - there should be an option to disable *ProScan's* virus removal capability to stop uninformed end-users compounding any viral damage. ProScan lags behind McAfee's shareware products in terms of both programming and currency. If the developer overcomes the various shortcomings highlighted in this review, *ProScan* might conceivably earn its place in the corporate environment.

PROSCAN

Product ProScan v2.24

Manufacturer McAfee Associates, 4423 Cheeney Street, Santa Clara, California 95054-0253, USA. Tel 408 988 3832, Fax 408 970 9727

Price	On Application
Memory Check	No
Network Aware	Yes
Single File Check	Yes
Definition Format	Proprietary
Virus Removal	Disinfection
Access To VB Test Set	No
User Upgradeable	No
Resident Scanner/Monitor	No

Scanning Speeds

Hard Disk 'Secure'	6 mins 56 secs
'Turbo'	3 mins 04 secs
Floppy 'Secure'	1 min 26 secs
'Turbo'	0 mins 54 secs

Scanner Accuracy

Parasitic	270 out of 365
Boot Sector	6 out of 8
Accuracy	73.99%

For an explanation of the entries in this table refer to the evaluation protocol published in *VB*, April 1991, pp. 6-8.

Postscript

London based company *International Data Security* has recently been appointed UK distributor for McAfee's shareware products. *IDS* does not currently market *ProScan*. Users who register their programs with *IDS* have access to a 24-hour Bulletin Board Service from which they can download the latest versions. *IDS* do not offer technical support. For information telephone (071) 631 0548.

REVIEW 2

Dr. Solomon's Virus Guard

More and more anti-virus software companies are offering memory-resident components as part of their product range. The very first of these was Ross Greenberg's *FluShot +* which first appeared as shareware some three years ago and became the basis for his *Virex-PC* commercial product. Now the list of companies offering virus-specific memory-resident programs includes *Central Point Anti-Virus*, *Norton Anti-Virus*, *Dr. Solomon's Anti-Virus Toolkit*, Fridrik Skulason's *F-Prot* and Bates Associates' *VIS Anti-Virus Utilities*.

Essential Criteria

From the user's point of view, the essential criteria which apply to these programs are:

- ➤ Is the utility offered both as a device driver and as a TSR? A device driver offers a greater level of security than a TSR but at the expense of not being compatible with most network shells.
- ➤ Does it make the best possible use of available system resources? For example, if expanded memory is available, does the utility use it to store code and/or data?
- ➤ Is its conventional memory footprint small?
- ➤ Is its presence unobtrusive under normal, clean conditions?
- ➤ Is it compatible with other memory-resident software?
- ➤ Is it secure and effective under all conditions?

It is against this background that I looked at *Virus Guard* which is a recent addition to *Dr Solomon's Anti-Virus Toolkit.*

Principal Components

Virus Guard (version 1.3 released June 3rd 1991) consists of four files: AUTHOR.COM, which 'stamps' diskettes with an authorisation code which can be checked by *Virus Guard*; GUARD.COM, the conventional memory version of *Virus Guard*; GUARDEMS.COM, a version which uses EMS; and, GUARD.DRV, which contains the recognition patterns used by either of the *Virus Guard* programs.

Virus Guard is a TSR program which can be loaded into memory either from the command line, through a batch file (AUTOEXEC.BAT for instance) or by a network login script.

Virus Guard defaults to checking files that have been opened for read access, programs that are about to be executed and the boot sectors of diskettes that are accessed. Boot sector checking and files opened for read access can both be disabled at load time, through command line options. Once *Virus Guard* is in memory, you cannot change its detection characteristics nor can you disable or unload it.

Experiments With 4K

Checking files during read operations is not a secure modus operandi as the monitor can so easily be circumvented by stealth viruses.

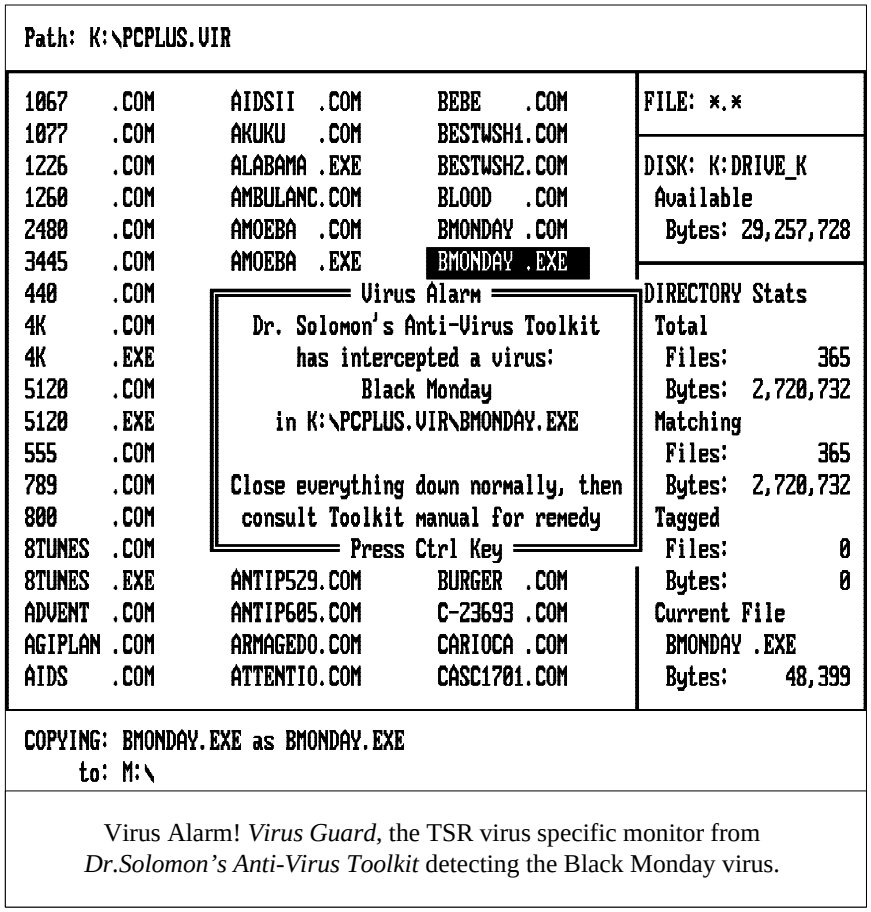
Virus Guard detected a copy of 4K attached to a file when it [*Virus Guard*] was started in a clean environment. However, on a machine where the 4K virus had infected COMMAND.COM and *Virus Guard* then became resident, 4K continued to infect program and data files undetected - indeed GUARD.COM itself became infected and still did not detect the virus.

Virus Guard completely failed to detect 4K when the virus was launched from a packed file. On a clean machine an infected copy of GUARD.COM was actually responsible for introducing the virus - again without alert.

This illustrates the major disadvantages of memory resident monitors which:

- ➤ are not device drivers (the infected COMMAND.COM would have been detected as it was loaded by DOS)
- ➤ Do not check memory upon loading for resident viruses (4K would then have been detected)
- ➤ Ignore disk-write operations

Aware of the potential security loopholes with loading a TSR as a batch file, *S&S* is currently developing GUARD.SYS, a *Virus Guard* device driver which can be run before COMMAND.COM is executed. This device driver will be available free of charge to registered users upon request.



Virus Alarm! *Virus Guard*, the TSR virus specific monitor from *Dr.Solomon's Anti-Virus Toolkit* detecting the Black Monday virus.

Detection Rating

Overall, *Virus Guard* was able to detect an acceptably high number of viruses and its 'hit' rate was found to be only marginally lower than *FINDVIRUS*, the *Toolkit's* disk scanner.

Among the viruses it failed to detect were Casper, Number One, Tequila, 1260, V2P6 and PCVRSDS. (In fairness, the documentation clearly states that *Virus Guard* will not detect V2P2 or V2P6.)

Virus Guard does not detect the Tequila virus either when it is introduced into a clean system, or after rebooting from an infected boot sector. *Virus Guard* did not detect the virus during its subsequent spread around the disk. Its failure to detect Tequila is somewhat unnerving considering the recent spread of this virus in the wild.

One point worth bearing in mind is that *Virus Guard* may well provide a different name to a virus than that provided by *FINDVIRU*, since the former product uses the same identification pattern to identify more than one virus. For example, it identified Monxla, Polimer and Turbo Kukac as 'Kukac' and Cookie, Machosoft and Syslock were all identified as Cookie.

Memory Footprint

Both versions of *Virus Guard* occupy the same amount of conventional memory - just over 5 kilobytes, but the EMS version allocated 32 Kbytes of expanded memory in order to store its

signatures (from GUARD.DRV). This made the EMS version less intrusive than the non-EMS version which had to refer frequently to its disk-based signature file.

Overhead

The conventional memory-only version (*Virus Guard*) adds approximately 25 percent to the time taken to copy ordinary files or load and execute programs - this increases to approximately 1000 percent if *Virus Guard* is loaded from diskette.

I suspected that not all program loads were being checked - this was confirmed by loading a DOS services 'spy' TSR program before *Virus Guard* which was loaded from a floppy diskette. From the results, I concluded that *Virus Guard* checked less than half of all the programs I ran. It randomly did not check every invocation of *Xtree*, *The Norton Utilities*, the text editor used to prepare this review and the *Windows* files WIN.COM and WIN386.EXE, among other programs.

Virus Alerts

When *Virus Guard* does detect a virus, a pop-up window displays the name of the virus; this is accompanied by a continuous wailing noise from the speaker. Pressing either of the Control keys stops the alarm and restores the screen. The screen message can be customised to suit user requirements - this could provide the name and extension number of a company's technical support department, for example.

I tested the TSR in most screen modes, both graphical and textual, up to and including VGA (640 x 480) and noticed no nasty glitches. I did notice that on several occasions, *Virus Guard* allowed DOS to retry the copy operation with the result that the warning screen was redisplayed, requiring a second (and sometimes, a third) press of the Control key.

TSR Compatibility

Virus Guard coexists well with *Borland's Sidekick* - both the popular original as well as later versions - and with the *Simon* TSR text editor. The EMS version was less well behaved when *QEMM* (v5.1) was used to provide the EMS services, but I suspect that it was *QEMM* that was the guilty party as I have noticed similar curious interactions concerning that particular version of *QEMM* and other products. In *Virus Guard*'s case, this manifested itself in a curiously high number of false positive alarms - where there were none with other EMS drivers - which suggests that *Virus Guard*'s expanded memory block had become corrupt.

Conclusions

Overall, *Virus Guard* will prove an acceptable product for use in low risk areas - that is to say, by non-networked users running standard applications who do not use modems and have limited exposure to 'foreign' diskettes. The main security loophole with this program is that even viruses that it 'knows' about can be introduced into the processing stream and once they gain control of the system, *Virus Guard* provides little protection against them. Inadequate self-checking mechanisms make it essential that a thorough initial integrity check using proven scanning programs is undertaken before *Virus Guard* is installed.

It should also be borne in mind that users will generally install software of this type within its own subdirectory. Under these circumstances, GUARD.COM is likely to be amongst the last files to be infected. Thus the fall-back self-test detection (if it can be called that) will only occur after most of the files on the disk have become infected.

Regarding load, execution and copying overhead (which was not a problem under normal operating circumstances) it seems that *Virus Guard* uses some method to analyse the files it is checking internally. When presented with files other than straightforward program code (e.g. executables packed with dynamic decompression utilities such as *DIET*, *LZEXE* or *PKLITE*), this analysis imposes noticeable overhead. With *Virus Guard* running from the hard disk, copy overhead for packed files was measured at an average of 120% - a figure which rose to an average 1008% when *Virus Guard* was invoked from a floppy drive.

No major problems were encountered with *Windows 3* compatibility when operating in an uninfected environment. However, in 386 enhanced mode, the machine froze when *Virus Guard* checked virus infected files during multiple *Windows* sessions.

Finally, *Virus Guard* behaved very well in the company of commonly used TSR programs.

Virus Guard Version 1.3

Virus Guard is the latest addition to *Dr. Solomon's Anti-Virus Toolkit* (version 5.11).

The developer and vendor of the program is *S&S International*, Berkley Court, Mill Street, Berkhamstead, Hertfordshire HP2 4HB, UK.

Tel 0442 877877, Fax 0442 877882.

A review of *Dr. Solomon's Anti-Virus Toolkit* appeared in *VB*, June 1991, pp. 18-19.

Evaluation Hardware

An Apricot Qi 486-25-320. This is a 25 MHz 486 MCA PC fitted with 16 MB of RAM and a 320 MB SCSI hard drive which was partitioned into 10 logical drives. Part of the extended memory was configured as a RAM disk thus providing drives A to M inclusive.

END-NOTES & NEWS

(IBM VIRUSES (UPDATE)

Spanz - CN: Infects files in current directory and on PATH (first file found when run). Six months after all suitable files have been infected, the volume label changed to 'INFECTED!' if run in first second of any minute. Virus ends with text '* SPANZ *'. Considers files infected if seconds field set to 0,16,32 or 48. Infective length is 639 bytes.

```
Spanz           8D9C 7D03 0683 BC76 0300 7415 8B84 7403
```

Witcode - ER: A 966 byte virus awaiting analysis.

```
Witcode         83FB 0473 088C C048 8EC0 83C3 1026 8B77
```

HLD Publishing Company of Los Angeles, USA, is advertising computer virus code for sale. The company's advertisement in the *Microtimes* computer magazine offers a fully-functioning Jerusalem virus for US$29.99. Meanwhile, Michigan based publishing house *Abacus* has released ***Computer Viruses and Data Security*** by Ralf Burger despite written warnings that the publication of source code in Burger's previous book *Computer Viruses: A High Tech Disease* is directly responsible for the appearance of numerous computer viruses. *Abacus* also distributes Burger's *Virus Secure for Windows* software. *Computer Viruses and Data Security* was released in the US on July 12th. *VB* intends to review the book in September. Information from *Abacus*, 5370 52nd Street SE, Grand Rapids, MI 49512, USA. Tel 616 698 0330, Fax 616 698 0325.

The UK *IT Security Evaluation & Certification Scheme* released **UK Certified Product List** issue 1.1 (UKSP 06) on June 1st 1991. Three anti-virus products have now been certified to UK Level 1: *Eliminator* (v1.17) from *PC Security Ltd*., *Norton Anti-Virus* 1.0.0 from *Symantec UK Ltd*., and *Vaccine* version 4.08 from *Sophos Ltd*. Information from Rm 2/0609, *CESG*, Fiddlers Green Lane, Cheltenham GL52 5AJ.

Virus Bulletin Conference, Hotel de France, Jersey, September 12-13th 1991. Contact Petra Duffield. Tel 0235 531889.

IBM UK is running two one-day **Virus 'Master Classes'**. The dates are September 16th (Manchester) and September 18th (London). Information from *IBM Customer Education*. Tel 0256 56144.

EICAR (*European Institute for Computer Anti-Virus Research*) is holding a two day **Virus Seminar** in Brussels, Belgium, September 24-25th. Tel Guenther Musstopf +49 40 6932033 or Dirk Giroulle +32 3 231 6308.

Sophos Ltd. is holding a one day seminar on **Anti-Virus Strategy for Software Producers** in London, November 19th 1991. Tel 0235 559933.

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