September 1992 ISSN 0956-9979

VIRUS BULLETIN

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

CONTENTS

EDITORIAL

MtE Detection - A Case Of *All Or Nothing*?

In recent months two distinct theories have emerged regarding the detection of viruses which employ Mutation Engine encryption.

Some practitioners believe that a scanner must detect every single instance of an MtE virus in any of its billions of possible forms. This group argues that the failure to detect just *one* virus infected file will result in the eventual re-infection of every suitable target file on the machine. For instance, in his attempts to guarantee a 100% detection rate of this class of virus, the Technical Editor of this journal recently generated more than 1 million MtE progeny against which he ran his scanner *F-PROT* (which passed this extraordinary test with flying colours). Similar tests conducted by Vesselin Bontchev at the *University of Hamburg* involved the generation of a more modest 20,000 MtE offspring in order to test those scanners claiming guaranteed MtE detection.

Other practitioners, possibly taking a lazier and more relaxed approach, argue that a 99% (or even lower) detection rate of MtE polymorphism is adequate. They argue that the appearance of two, five, fifteen or a hundred MtE infected files on any given platform is sufficient indication that the afflicted machine requires a fundamental ‘clear out’ i.e. the deletion of every single executable file and a thorough restoration from clean masters.

Those who argue in favour of ‘guaranteed’ MtE detection are arguably more conscientious than their less pedantic brethren. Certainly, non-specialists can hardly be expected to understand the subtle intricacies of ‘polymorphic’ computer viruses or ‘MtE encryption’ and keeping naïve users out of harm’s way by guaranteeing to detect such beasts in all their horrible guises must be a Good Thing. The only question which remains is whether one million MtE generations is sufficient to guarantee the detection of a virus which replicates in many *billions* of possible forms?

Taking the issue of polymorphism one stage further, it should be remembered that the Mutation Engine is not *itself* a virus but an encryption engine with which to *conceal* a virus. There are currently a number of viruses from a diversity of sources which employ MtE camouflage, including Pogue, Dedicated, Fear and Groove. The practitioners are once again divided; this time the argument surrounds the issue of the *exactness* of identification. Some researchers believe it is necessary to decrypt the MtE camouflage sufficiently to identify the precise identity of the virus beneath it. This is analogous, perhaps, to stripping wallpaper or peeling an onion. The reasons propounded are that such precise identification will alert the user to the presence of any pernicious trigger effects, the possibility of corruption or other damage particularly associated with the specimen identified beneath its MtE shroud. Other researchers argue that the accurate identification of MtE encryption *itself* is sufficient to alert the user that something is awry and that prompt action should be taken.

As the concept of ‘polymorphism’ becomes more widely known within the virus writing fraternity, it seems probable that the ‘lazy’ camp’s theories will prevail. The Mutation Engine will inevitably turn up soon in its source code form and the hackers will get to work on those all important ‘improvements’. In the fullness of time other encryption engines will spring up as if from nowhere, and the addition of sophisticated ‘polymorphic’ routines will become standard practice. All of which should add enormously to the overheads of your friendly anti-virus software developer - so remember: he’s not lazy, just *overworked*...and clever and modest and misunderstood!

Stop Press: Man Charged With ‘Malicious Mischief’

On August 20th 1992 former Canadian computer magazine publisher Richard Brandow, 28, was charged with planting a computer virus which infected thousands of copies of *Aldus Corporation’s Freehand* software in 1988.

Brandow, who now writes for the ‘Star Trek’ television series, has been charged by prosecutors in King County, Washington, USA, with malicious mischief and could face up to 10 years’ imprisonment if convicted.

Brandow says he finds the charge surprising. “What are they going to do?” he asked, “It happened four years ago and I am here in Montreal.” Brandow told *Associated Press* that the virus, which infected Macintosh computers, wished a universal message of peace on March 2nd 1988 and then removed itself from the system folder. Brandow, who left his name in the screen message, claims that the program was educational in its intent.

The virus infected a master disk of the *Freehand* illustration package at *Aldus* which subsequently had to recall 5,000 copies of the program and destroy a further 5,000 disks held in inventory. The incident cost *Aldus* an estimated $7,000. Ivan Orton, King County deputy prosecuting attorney said that this was the first time that the state of Washington had filed such criminal charges. This is the latest in a spate of indictments against programmers in the United States - in June of this year two *Cornell University* students were charged with distributing a Macintosh Trojan horse to bulletin board systems (see *VB*, August 1992, p. 28).

TECHNICAL NOTES

Virus Construction Tools

Perhaps the most significant event in the past few months has been the long-anticipated arrival of functional virus construction toolkits. Today two such toolkits are in circulation - VCL, which was briefly described last month, and PS-MPC, which has just appeared. The programs are different - VCL has a full-screen menu system, while PS-MPC requires the user to edit a configuration file to create a new virus. However, the purpose of both programs is the same: to enable users with little or no programming knowledge to create viruses quickly and easily.

The spectre now haunting researchers is the arrival of a diskette in the mail with a thousand or so new viruses, created in a single weekend using one of these toolkits.

Construction Sets - A Short History

Virus construction tools are not new. The first known virus toolkit appeared in 1990 and was called VCS, or Virus Construction Set. This program generates a new virus each time it is run. However, there are no code differences at all between any two viruses generated by VCS. All viruses generated are 1077 bytes in length and all can be detected with a single scan string. The virus creator needs absolutely no knowledge of 80x86 assembly language to use the VCS program. This program has spawned only one well-known variant called Manta.

The Mutation Engine

The second virus ‘toolkit’ was CrazySoft, Inc.’s Dark Avenger Mutation Engine (MtE) which was described in *Virus Bulletin*, April 1992, page 11. This Bulgarian encryption engine allows virus authors to endow viruses with an almost limitless number of decryption routines. Although the virus creator needs to know how to write 80x86 assembly code, no knowledge of the inner workings of MtE, save the entry parameters, is needed to use this toolkit. It has been used to encrypt several viruses, including Dedicated, Pogue, Fear, and Groove. In truth, the Mutation Engine is rather less than a full toolkit, being designed to disguise virus code rather than generate it.

Virus Construction Laboratory

A further ‘production line’ virus toolkit (or generator) to be released was VCL, or Virus Construction Laboratory. This was written by Nowhere Man of the NuKE virus writing group based in California. This toolkit allows the user many options, including the creation of parasitic COM infectors, companion EXE infectors, and the ability to insert Trojan horse and logic bomb routines into the code. VCL can only handle parasitic infections of the COM file format and it incorporates only one decryption formula. Furthermore, the initial release included a quirky installation program which fails to install properly under certain conditions. This toolkit contains an integrated development environment (IDE) loosely based on the *Borland* interface. This IDE is simple to use and non-programmers can understand it without knowledge of 80x86 assembly language.

PS-MPC

The latest addition is the PS-MPC - the Phalcon/Skism Mass-Produced Code Generator 0.90a created by ‘Dark Angel’. This tool generates viral code according to user-designated specifications. The output is in TASM-compatible Intel 8086 assembly language and it is up to the user to assemble the output into working executable form. The PS-MPC include the incorporation of an encryption technique which resembles that found in the V2P6 virus (whereby one decryption algorithm uses randomly chosen registers), COM and EXE file infection, and critical error handling.

How Dangerous Are The Simple Toolkits?

Do these two new ‘production line’ or ‘generator’ toolkits represent a serious threat? The viruses they create are currently fairly simple, non-resident infectors, but the authors of both packages promise the ability to create resident viruses in future versions.

The viruses do not use any ‘stealth’ techniques, but are generally encrypted. The encryption techniques are simple, slightly polymorphic, but detectable with a set of search patterns. However, it is important to note that although each individual virus may only use a primitive, fixed, encryption method, the encryption may (especially in the case of the PS-MPC) vary from one virus to another.

In other words, it is possible to detect each virus produced by the toolkits with a search pattern, but producing a generic search pattern which will detect all of their possible progeny is impractical. This is particularly the case because the virus creator can modify the assembly output slightly before releasing the virus - for example by adding a few ‘garbage’ instructions to the decryption routine.

Two classes of anti-virus products have done very well in detecting the VCL and PS-MPC created viruses. The current heuristic scanners detect all the viruses they have created, while checksumming programs will easily detect the changes they cause.

Compressed Replication

Viruses are frequently distributed in a compressed form - where the executable has been packed by *DIET, PKLITE, LZEXE* or any other dynamic decompression program. The benefits to the virus distributor are obvious - unless a virus scanner is actually able to scan inside compressed files, it will miss the original sample on disk, although it can easily detect subsequent generations.

A few viruses are also distributed in a compressed form, but with one significant difference - later generations of the virus are also compressed. Such a virus replicates in a unique way - when infecting files, it cannot simply write its own memory image to the file, because it has been decompressed. Instead, the virus, once resident, must *read* the compressed file containing itself into memory and write this *disk image* back to the target file. As the compressed file cannot be appended to the target file, viruses of this type are limited to three classes - overwriting, COM file-prepending or companion-type.

'Degeneration'

In practice, most viruses cause corruption or system malfunction of some kind. This is usually the result of poor design or poor coding, but another factor to be considered is a theory proposed by some researchers which might best be termed 'degeneration'. The theory, touted some years ago as the 'mutation threat', maintains that errors introduced by the constant copying of viral code would inevitably result in new, more dangerous viruses.

There are occasional tiny changes within virus specimens which make the phenomenon of degeneration interesting. A particular example has been noted in the V-SIGN virus (see pages 16-18). The specimen disassembled for this analysis has several errors within the code but one particular mistake is not matched in a sample from another source. It may well be correct to ascribe this one bit error to degeneration, i.e. a corruption which has occurred at some stage during the replication process across hundred or thousands of disks.

Another False Positive

A hexadecimal pattern published for the Penza virus in the August 1992 edition of *VB* caused a false positive when used in conjunction with the *VIRSCAN* detection program from IBM. The false positive occurred in the DOS FORMAT.COM program when *VIRSCAN* was run with its 'mutants' option enabled. If *VIRSCAN* is updated with *VB* patterns it should be run in its 'no mutants' option. A replacement pattern follows:

```
Penza  BE64 04B8 00FF CD21 3BC8 7503
       E988 00FC 8CC8 4850 0726 803E
```

A Million MtEs

The obvious approach to check a scanner which claims to detect polymorphic viruses such as those using the Mutation Engine is to generate a large number of samples and to see whether the scanner misses any of them. *VB's* Technical Editor recently checked his own scanner in this way, generating and identifying slightly more than one million MtE samples in an overnight test.

To generate one million samples, a program was written to create files which employed the MtE but which were not themselves viruses. The generator program accepted a command line parameter which stated the number of files which should be created. A batch file ran this program, creating 255 files on each round. The scanner was then run and instructed to delete the MtE encrypted files as they were detected. Any file which was not detected (none, as it turned out) were then to be copied to a different file.

To speed this process the test was conducted on a fast PC (a 50 MHz '486) and all files were stored on a RAM disk. The scanner was invoked with self-test and screen output disabled. Each iteration took less than 11 seconds, enabling one million generations to be checked in 12 hours.

The test does not prove that this particular scanner is guaranteed to detect the MtE - it only indicates that the chance of missing an infection is very low.

Virus Prevalence Table - July 1992

Incidents reported to *VB* in the UK during July 1992

Virus	Incidents	(%) Reports
Form	8	22.8%
Spanish Telecom	7	20%
New Zealand II	6	17%
Eddie II	2	5.7%
Cascade	2	5.7%
NoInt	1	2.8%
Joshi	1	2.8%
Helloween	1	2.8%
Disk Killer	1	2.8%
4K	1	2.8%
Music Bug	1	2.8%
Flip	1	2.8%
Vacsina	1	2.8%
1575	1	2.8%
Tequila	1	2.8%
Total	35	100%

IBM PC VIRUSES (UPDATE)

Updates and amendments to the *Virus Bulletin Table of Known IBM PC Viruses* as of 23rd August 1992. Each entry consists of the virus' name, its aliases (if any) and the virus type. This is followed by a short description (if available) and a 24-byte hexadecimal search pattern to detect the presence of the virus with a disk utility or preferably a dedicated scanner which contains a user-updatable pattern library.

Type Codes

C = Infects COM files **E** = Infects EXE files **D** = Infects DOS Boot Sector (logical sector 0 on disk)

M = Infects Master Boot Sector (Track 0, Head 0, Sector 1) **N** = Not memory-resident

R = Memory-resident after infection **P** = Companion virus **L** = Link virus

Seen Viruses

66A, Satan-B - CN: The name '66A' is misleading, as the virus is 512 bytes long. It appends itself to COM files. Awaiting analysis.

```
66A              2D03 008B F581 C6FA 0189 4401 8BF5 8BFE 81C6 C401 81C7 FD01
```

Ash-743 - CN: A revised variant of the Ash virus reported last month. Much longer than the original, most of the code comprises silly text messages.

```
Ash-743          8DB6 0401 BF00 01B9 0400 FCF3 A4B4 1A8D 96EB 03CD 21B4 4E8D
```

Athens - CER: This 1463 byte encrypted virus originated in Greece. It contains the text 'TROJECTOR II,(c) Armagedon Utilities, Athens 1992' but it is totally unrelated to the Armagedon virus.

```
Athens           061E 5053 51E8 0100 ??5D 83ED 0890 FC0E 1FBE 2800 03F5 8BFE
```

Chad - CN: This 751 byte virus was discovered in the wild. It contains several text strings, including 'WOT!! No Anti-Virus'.

```
Chad             8904 B803 002B F889 7C02 8B44 0489 058A 4406 8845 028B D6B8
```

Dad - CR: A minor variant of the Suriv 1.01 virus, with no changes to the code, but the text messages translated to Spanish. Probably written in Argentina. Detected with the Suriv-1.01 pattern.

Enet 37 - CN: This is a variant of the South African Friday the 13th virus, but it is 613 bytes long. Detected by the Friday the 13th (South African) pattern.

Flower - EN: This virus activates on November 11th (any year), overwriting infected files with a Trojan that displays a text message.

```
Flower           2E8C 1EA5 038C C88E C08E D880 3E02 0090 7416 8A16 0200 BB36
```

Freddy - CER: This 1870 byte virus has been reported in Brazil. It is awaiting analysis, but is known to contain destructive code. The virus contains one encrypted text string: 'Freddy Krg'.

```
Freddy           070E 1F01 0610 0001 0614 008C 0616 008C 0604 008C 0608 008C
```

Frodo-D - CER: Virtually identical to the original 4096 byte (4K) virus, but with a single instruction changed - probably to avoid detection by a particular scanner. Detected with the previously published 4K (Frodo, 4096) pattern.

Fungus - CER: An Australian stealth virus, which contains several text strings, such as 'X-Fungus by Harry McBungus' and 'Epilectic Downer'.

```
Fungus           803E 0000 5A74 0A40 0306 0300 3D00 A072 ED8E D88B D8A1 0300
```

Futhark - CR: The Futhark virus is a more advanced version of the Youth virus, with limited stealth abilities. It is encrypted, but the decryption key appears to be 0 in all samples which have been generated, making it possible to select a pattern from the body of the virus, instead of from just the decryption routine. As a precaution a second pattern, extracted from the decryption routine, is provided.

```
Futhark          80FC 1274 BB80 FC4E 74B9 80FC 4F74 B42E 803E 9501 0074 03E9
Futhark-decrypt B9AD 03BE 1B01 89F7 AC34 ??AA E2FA
```

Horror-1112 - CER: Closely related to the other two Horror variants reported earlier.

```
Horror-1112      8BFE 83C7 0AB9 0A04 2E8A 8456 042E 3005 FEC0 47E2 F8C3
```

Jerusalem-Timor - CER: A 1562/1567 byte variant which contains several text strings. It activates on November 12th (any year), when it displays the message ‘St. Crus, Dili’. Detected with the previously published Jerusalem-1735 pattern.

Made - CN: This 334 byte encrypted virus contains the text ‘Made in England’, which might be true, although the first sample arrived from California. Does not seem to contain a payload or side-effects.

```
Made           9009 09E8 0000 5E56 81EE 0B01 E809 00E8 0001 EB30 90
```

MtE-Cryptlab - CN: A new virus using the Mutation Engine - programs already able to detect MtE should be able to detect this variant without any problems.

Npox - CER: 963 bytes long and contains the text ‘Evil Genius V2.0 - R.S/NuKE’. The virus is destructive and may trash the disk on the 18th of any month.

```
Npox           3D00 4B74 1780 FC11 74AE 80FC 1274 A93D CD7B 7503 EB06 902E
```

Otto6 - CN: A primitive virus written by a new group of virus writers which calls itself ‘YAM’ (Youngsters Against McAfee.)

```
Otto6          E800 005E 5681 EE08 0158 2D00 01A2 FF00 56B9 6002 81C6 2501
```

Parity Boot - MDR: This virus, which is probably from Germany, intercepts INT 9H (keyboard interrupt), and may randomly display the text ‘PARITY CHECK’ and hang the computer.

```
Parity Boot    CD19 80FC 0275 4783 F901 7542 80FE 0075 3D50 5356 579C 2EFF
```

Quake - ER: This 518 byte virus from The Netherlands which has a very noticeable screen-effect - a section of the screen visibly shakes when the virus triggers.

```
Quake          80FC 4E75 722E FE0E 0602 7466 5053 5152 1E06 0E1F B824 35CD
```

Rust - CER: A 1710 bytes virus which is probably of Russian origin.

```
Rust           0E5A 8EDA 8EC2 BE6D 018B FEB9 B502 E8B5 FA5A 071F C3B0 03CF
```

Screaming Fist-732 - CER: Another member of this family. There are no significant differences from previously reported variants.

```
ScreamF-732    5D8B F556 B0?? B9C7 02?? 2E30 0446 E2F9 C3
```

StinkFoot-D - CN: A 1273 byte variant, probably by the same author as earlier versions. Exceptionally badly written and works only on ’286 machines and above.

```
Stinkfoot-D    600E 59BA 0400 B435 B024 CD21 061F 8957 0289 0F61 1F07 C3BE
```

Sux, Anti-MIT - CN: A 770 byte virus which activates on December 1st (any year), when it displays the message ‘MIT Sux!’ and attempts to trash the hard disk.

```
Sux            BE2A 018A 2607 01EB 1290 AC32 C4AA E2FA B419 CD21 8AF0 B40E
```

Swiss Phoenix - CER: This 927 byte virus was reported ‘in the wild’ in Switzerland. It contains one encrypted text string: ‘Phoenix’. Awaiting analysis.

```
Swiss Phoenix  FF74 2BA1 2C00 501F BA08 00B8 003D CD21 721B 8BD8 2EC6 8444
```

Trivial-26 - CN: Yet another attempt to write the world’s smallest virus. As a full-size pattern would contain almost the entire encoding of the virus, only a short pattern is published here, which can be found at the beginning of infected files.

```
Trivial-26     2A2E 2A00 B44E 89F2 CD21 B802
```

Trivial-50 - CN: Another small, overwriting virus, which does nothing but replicate.

```
Triv-50        B44E B927 00BA 2C01 CD21 7207 E806 00B4 4FEB F5CD 20B8 023D
```

Trivial-Hanger - CN: One of the most obvious viruses ever, which has practically no chance of spreading whatsoever. It overwrites the first 143 bytes of infected files. When run it displays the message ‘System Hanger! Enjoy!’ and subsequently ‘hangs’ the system.

```
Hanger         5BCD 21B4 4FCD 2173 DEBA 3B01 B409 CD21 FAF4 CD20 2A2E 432A
```

VCL: Three viruses have been created to date using the Virus Construction Laboratory (see *Technical Notes*, pp.3-4 and *VB*, August 1992, pp. 3-4). Different search pattern(s) must be used for each group.

VCL Encrypted - CN: (Venom, ENUN, Code Zero, Kinison, Earth Day and Donatello)

```
VCL-1          8DB6 0E01 B9?? ??81 34?? ??46 46E2 F8C3
VCL-2          8DBE 0E01 B9?? ??81 35?? ??47 47E2 F8C3
```

VCL Non-encrypted - CN: (Yankee-Tune)

```
VCL-3          ACB9 0080 F2AE B904 00AC AE75 EFE2 FA89 BF?? ??8C 87?? ??C3
```

Youth - CR: Awaiting analysis.

```
Youth          89D7 01CF 4F8A 058A 5DFF 8845 FF88 1DE8 B600 3D00 4B75 7C2E
```

ZZ - CN: A 429 byte virus which does not seem to do anything but replicate and occasionally display the text ‘ZZ Top the best !!!’.

```
ZZ             CD21 BA89 01B9 2100 B44E EB7E 8BFE AC0A C074 0534 CAAA EBF6
```

FEATURE

Dr Richard Ford

Oxford University's Virus Hunters

In March 1991 *Oxford University Computing Services* (OUCS) realised that it was under siege.

The Spanish Telecom computer virus, a highly destructive specimen believed to have been written on the 23rd August 1990 by the self-styled Grupo Holokausto in Barcelona, Spain, had spread unnoticed and at an alarming rate throughout the University's departments.

Seventeen months have passed since the simultaneous discovery of the virus at *Oxford University* and at *City University* in London. In August of this year, *VB* returned to the dreaming spires of Oxford to examine the subsequent changes introduced by OUCS and the lessons learned from this outbreak.

Micro Advisory

Oxford University has a large number of computers of all types scattered throughout its many and varied departments and colleges. *Oxford University Computing Services* (OUCS) acts not only as a centre for the University's VAX cluster, but also as a help area and advisory service for any user having difficulty with their machine.

If you contact OUCS and report a virus problem, the person you will probably speak to is Lynne Munro of Micro Advisory. While contacting her to arrange an interview I explained that I wanted to ask her about virus prevention in Oxford. “Impossible!” was the instant reply; after seventeen months and a constant barrage of publicity she is still receiving diskettes infected with the Spanish Telecom virus.

Micro Advisory is involved with all aspects of PC support, virus hunting being but one of its many duties. During our meeting the telephone rang several times with questions about graphics packages, disk compatibility and other software and hardware enquiries; Munro is obviously accustomed to frequent interruption.

It seems strange that after all the warnings about the Spanish Telecom virus, raised both by OUCS and the local newspapers and radio, that the virus should still be active in the University, although very much less prevalent. As Munro proceeded to explain, a number of factors combine to make virus suppression in the ancient seat of learning an up-hill battle.

Computer Misuse Act

Spanish Telecom was the first major virus outbreak at *Oxford University*. In Autumn 1990, OUCS was alerted by Dr Peter Lammer of *Sophos* that the virus had been found on a PC within the OUCS help area. In the following months the scale of the problem became clear; students and staff started noticing strange disk errors and staff at OUCS soon recognised similarities between a catalogue of reports.

Munro, realising an offence had taken place under section 3 of the *Computer Misuse Act 1990*, contacted *Thames Valley Police* to report the virus. There was some confusion at the force's HQ in Kidlington, just outside Oxford as to how to deal with the situation and even as to exactly what crime she was trying to report. Eventually, after speaking to a number of bemused officers, she was put through to the data manager who gave her the number of the *Computer Crimes Unit* based in London. “Everybody was very helpful at this time,” recalls Lynne, “especially *Sophos* [based in nearby Abingdon] and the rest of the anti-virus community. I think before Spanish Telecom we didn't really know the scale of the problem.”

“Education is where OUCS can help most I think. Given the structure of the system what else can we do?”

One of the many dangers associated with the virus was that the scanner in use by OUCS (*McAfee's SCAN*) did not, at the time, recognise the virus. The alert was put out to the departments. The most effective communication channel available to OUCS is its network of computers, in particular its VAX cluster. It was only when the Computing Services was finally able to scan for the presence of the virus that the true extent of the infection became apparent: the virus had spread to virtually every computer-using department.

Spanish Telecom is a multi-partite virus which infects programs and the Master Boot Sector. If the infection is noticed before it executes its destructive payload, machines can be disinfected with relative ease. Some software was obtained to do this task, and made available via OUCS.

Fragmentation

The virus hunt commenced. It soon became evident that the highly distributed structure of *Oxford University*, which has no single campus but which is comprised of more than thirty independent colleges and a similar number of academic departments, was a major hindrance to the detection and disinfection process. It is this fragmentation which has made it an almost impossible task to eradicate Spanish Telecom completely.

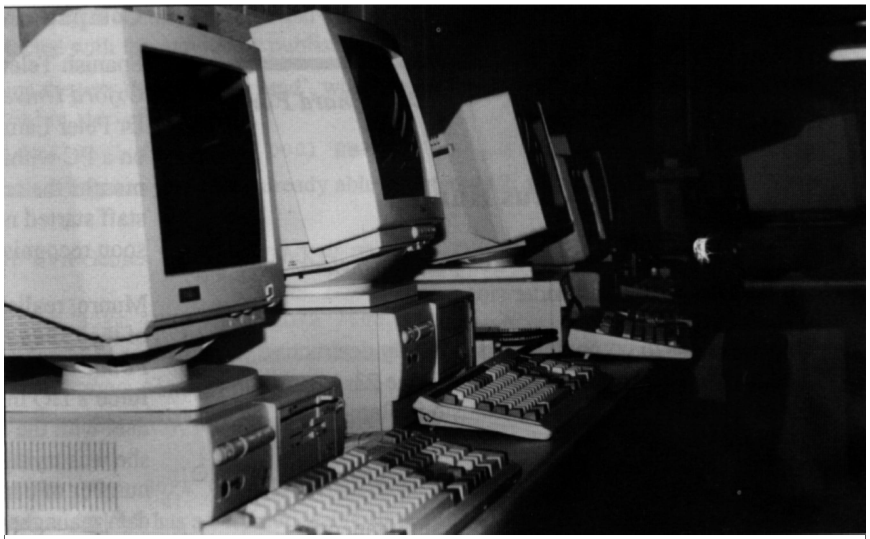
This multitude of departments, which frequently break down into subdivisions of research groups and even individual researchers with their own computer, represents an organisational nightmare for OUCS which is an advisory body for all matters computational.

The individual colleges frequently provide communal computing facilities for their students and fellows. These 'in house' facilities tend to be managed by computer enthusiasts with no official post. It is a somewhat permissive environment with a large amount of disk swapping; the ideal breeding ground for computer viruses. The difficulty for OUCS is in reaching everyone - if just one disk remains infected, the virus is reintroduced into the processing stream. "Education is where OUCS can help most, I think." Munro says, "Given the structure of the system what else can we do?"

Countermeasures

Certainly OUCS has taken all the correct steps to help alleviate the problem. Software is readily available to members of the University, posters abound in OUCS, and screen messages displayed immediately after logging on to the VAX give details of the current situation.

OUCS negotiated University-wide site licences for *Sophos*' *Sweep* and *McAfee's Validate*, joined the *VIRUS-L* electronic conference and subscribed to *Virus Bulletin*. Help is available any weekday during office hours via the University's own internal telephone system and over outside lines. The infrastructure thus exists to support scanning, checksumming and disinfection should any virus be discovered.


An enormous range of platforms and a fragmented structure consisting of more than thirty independent colleges and a similar number of academic departments combine to make virus suppression at *Oxford University* a daunting task.

In spite of these efforts, users still remain ill-informed about the virus threat and new outbreaks are a regular occurrence. The list of reported infections grows; Spanish Telecom is well known but other specimens are also encountered including New Zealand II, Form, Yankee Doodle, the SVC series, Jerusalem B, Italian, and DIR II.

Apathy

Terry Hastings, provider of Computer Support for the Physics Department, comments "People say 'It will never happen to me'... until it does." Hastings is responsible for over three hundred computers scattered throughout the Physics Department, which itself is broken up into sub-departments and individual groups. "People won't take the time to scan new disks brought into the system. We could buy a machine dedicated to virus scanning and make it freely available, and within a week nobody would use it." This apathy, noticeable amongst many users, is a major stumbling block for the virus hunters.

One of the most effective measures that Hastings uses is simply to take a copy of the boot sectors of as many machines as he can - in the event of a boot sector virus outbreak, the PCs can be rapidly restored. Again, the structure of the department makes it difficult to keep track of which computers are where. With many experiments taking place in the department, equipment is bought or moved around frequently, complicating the task further.

Hastings, in common with almost all the staff providing computer support, is only too happy to help explain and set up preventative measures. Ultimately, however, he relies on user compliance. For instance, one of the drawbacks of checksumming files in an environment such as the Physics Department is that the configuration of machines is always changing, often on a daily basis. When so much software is being written or where different, incompatible drivers are required for custom cards within PCs, the time taken for the integrity check is too great for many impatient users. The classic trade-off between providing security and convenience is ever-apparent.

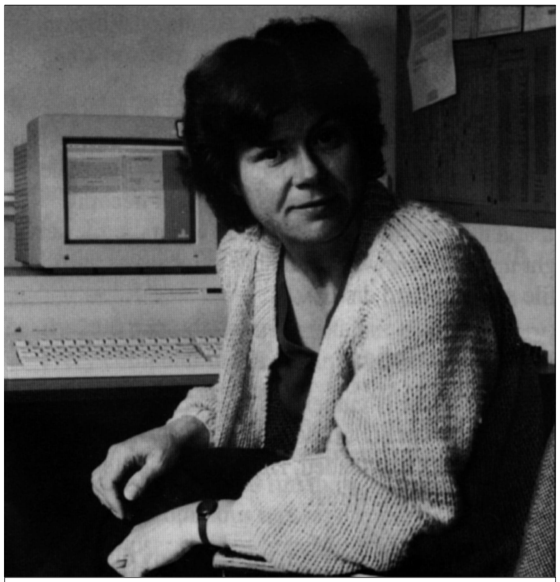
Michelangelo

Things are improving. The latest large-scale virus scare was posed by the Michelangelo virus. When asked about Michelangelo, Lynne Munro replied "The media coverage that the Michelangelo virus received helped a lot as it enabled us to reach people that we wouldn't normally reach. We put a note in the VAX login sequence, but very often, the people who are in contact with OUCS via the VAX are the people who already scan their own disks and take their own precautions."

Reaching the less sophisticated computer user can be a problem; naïve users tend only to contact OUCS when things go awry. Due to the media interest and early warnings about the Michelangelo virus, OUCS received only two or three isolated reports of machines infected by it, and the afflicted PCs were disinfected prior to the trigger date of March 6th.

C:>DEL *.*

Problems remain despite an intensive and ongoing education program. OUCS has had its share of pseudo-virus infections; the "Incidentally, I typed DEL *.* in the root directory" scenario is well



Munro: "We find ourselves recovering data from about ten machines every fortnight - that takes a lot of time".

known to its staff. Such naïvety and ignorance invariably shows itself when inexperienced users grapple with the mysteries of checksumming. "One thing that some users forget is that a fingerprinting program will only tell you that something in a file has changed, it doesn't necessarily indicate the presence of a virus," Lynne explains. "Now if anything goes wrong with somebody's computer they assume that it's a virus."

Novices and Wizards

Another aspect of *Oxford University* which further complicates OUCS's role is the varying level of computer literacy encountered, ranging from the total novice right through to the computing 'wizard'. OUCS runs a selection of courses which cater for all levels of computer literacy - the virus threat and the vital importance of data backups are now being emphasised heavily in the introductory courses. Lynne Munro believes that the real way forward is to combine all aspects of data security and protection: "We find ourselves recovering data from about ten machines every fortnight - that takes a lot of time."

A Shifting Population

The courses run by OUCS are having an effect, but to work properly the problems must be tackled at a university-wide level. OUCS is responsible for safeguarding a huge diversity of machines and, because of the structure of the system, lacks the powers that many PC support teams have. A shifting student population, open access PCs with multiple users, foreign students introducing exotic new viruses from their home countries, and a large population having access to ftp sites worldwide via *Internet*, are other factors which compound the department's difficulties.

The DIR II virus was brought in on a disk from a graduate student who lives in Cyprus. The infected file was detected on the hard disk of a PC in use at a communal computing room. Fortunately, the infected file had not been executed and the virus had not had the opportunity to propagate.

Oxford University has visitors from all over the world and the appearance of computer viruses previously unknown in the United Kingdom is a constant threat. "We need more communication from the rest of the University in order to get an idea of the scale of the problems. Many of the more experienced users don't inform us of infections which they've dealt with themselves" says Munro. "If people would contact us more often, we would at least know which viruses are out there." Many infections are taken care of at a departmental level, as each department seems to have acquired its own computing 'wizard'.

The battle is being won. The future is not so gloomy: "As the number of virus hits increases so does the wariness within the department," explains Terry, "it all reaches some sort of liveable level. We've learned a lot, and we're better able to cope." And Lynne Munro's conclusion? "We'll manage."

TECHNICAL EXTRACT

[*The following extract is from a paper written and researched by Morgan Adair of Novell Inc. and presented by Dominic Storey, Technical Account Manager of Novell UK at the 2nd International Virus Bulletin Conference, Edinburgh, September 2nd-3rd.*]

Morgan Adair
Technical Consultant
Novell Inc., Provo, Utah, USA.

Detecting Viruses in the NetWare Environment

Two classes of viruses present a particular threat to networks: viruses which infect boot sectors [DOS Boot Sectors] or partition tables [Master Boot Sectors], and viruses which infect program files.

This extract prescribes actions network managers can take to prevent the spread of viruses on their networks and how to limit the damage viruses can do. The extract also describes a NetWare Loadable Module (NLM) that can detect the symptoms of a virus infection on a *NetWare* file server.

Boot Sector Viruses

Viruses which infect boot sectors of floppy disks, the Master Boot Sector (Track 0, Head 0, Sector 1) and the DOS Boot Sector of the active partition of hard disks pose a unique threat to network file servers. A typical boot sector virus propagates in the following manner:

1. The computer is booted from an infected floppy disk and the virus in the disk's boot sector is thus executed.
2. The virus hooks an interrupt, then terminates and stays resident (TSRs), so it can infect other disks as they are accessed.
3. The virus executes the normal boot code from a copy of the boot sector it has saved.
4. When a disk is accessed, the virus copies the original boot sector (on a floppy disk) or the Master Boot Sector or DOS Boot sector (on a hard disk), then copies itself in place of the original boot sector.

A common boot sector viruses is NoInt, or Stoned III. It works in much the same way as the New Zealand virus, except that it does not display a message at boot time.

NoInt propagates in a similar fashion to most boot sector infectors:

1. The computer is powered on with a NoInt-infected disk in the floppy disk drive (NoInt is in the boot sector of the infected disk).
2. The BIOS boot program loads NoInt into memory and executes it.
3. NoInt installs itself at the top of memory and decreases the apparent amount of system memory by subtracting 2,048 bytes from the value at address 40:13H in low BIOS memory.
4. The virus hooks interrupt 13H, the BIOS disk I/O services, so that other disks can be infected as they are accessed.
5. NoInt executes the normal boot code from a copy of the boot sector it has saved on the infected disk.
6. The virus infects the first hard disk (if any) by copying the disk's Master Boot Sector to Track 0, Head 0, Sector 7, and then copies itself to the Master Boot Sector's rightful location (Track 0, Head 0, Sector 1).

Network Threat

A boot sector virus cannot spread from a workstation to a *NetWare* file server, for two reasons:

➤ *NetWare* volumes do not have boot sectors.

➤ Programs running on a workstation cannot call the low-level functions that read and write sectors on a file server's hard disks.

However, if the computer being booted from an infected floppy disk (containing a Master Boot sector infecting virus such as New Zealand or Michelangelo) is a *NetWare* file server, **the server's Master Boot Sector will become infected** and may be damaged such that when the file server is brought up, *NetWare* will be unable to locate its partition on the file server's hard disk.

File Server Prescriptions

A *NetWare* file server is vulnerable to attack by Master Boot Sector viruses at boot time, and when DOS is running before SERVER.EXE is loaded. Here are some steps you can take to minimize the threat:

☞ Use a virus scanning program on all floppy disks before inserting them into the server's floppy disk drive.

☞ Place a sticker over the file server's floppy disk drive with a message on it reminding users to scan all floppy disks before inserting them in the disk drive.

☞ Always boot from the same disk.

☞ If you do not need a DOS partition on the hard drive, do not have one. (This reduces the temptation to run DOS on the file server machine 'just to copy a few files' before bringing up the server.) Boot from a write-protected floppy disk that stays in the floppy drive (copy SERVER.EXE, STARTUP.NCF, and the disk driver to the boot disk).

Parasitic Program-Infecting Viruses

A network can be an effective vehicle for spreading a program-infecting virus (including the 'program' versions of multi-partite viruses). The following is a typical scenario for the infection of a network by a program-infecting virus:

1. A user receives a copy of the demo version of a new file utility from a friend. The program has been infected by a parasitic virus.
2. The user logs in and runs the infected program.
3. The virus code is loaded into memory along with the infected program and executes first. It places itself resident in memory, hooks interrupt 21H, then allows the utility program to run.
4. The user executes *MAP* and *WordPerfect*, but the virus is unable to infect the program files, because the user does not have Write rights to the directories in which they reside.
5. The user takes a break and plays a game another user has copied into a directory to which all rights have been granted to the group EVERYONE. The virus infects the game program.
6. Another user plays the game during the lunch hour. The virus, now resident in the memory of his machine, infects several utility programs on his local hard disk and the scheduling program used by everyone in the department (he was granted all rights to the directory containing the program, so that he could install and configure it).
7. By 2:00 p.m., the virus is resident in the memory of every computer in the department (except the Macintoshes), and has infected COMMAND.COM on every boot disk. At 3:00 p.m., the system administrator starts receiving phone calls. Two users' computers lock up every time they try to run the scheduling program. One user had an ominous message appear on her screen. Another user gets an 'insufficient memory' message every time he tries to run WordPerfect, although it ran just fine this morning. The system administrator logs in as SUPERVISOR and tries to run the scheduling program. It runs just fine for him (meanwhile, the virus has gone resident in his computer).
8. The system administrator runs *WordPerfect*. It also runs okay (although it may not next time, now that he has infected it).
9. Remembering the ominous message, the system administrator begins to suspect a virus, so he maps a search drive to a directory containing a virus scanning program. He runs the program, telling it to scan the entire file server directory structure (note that he has infected both the *MAP* utility and the virus scanner in the process). Fortunately, the virus scanner recognizes that its own program file has been infected, and that there is a virus resident in memory. The scanner tells the system administrator to reboot with a write-protected DOS disk, then run a separate program to remove the virus from all infected files.
10. The system administrator spends the next two hours scanning and removing the virus from infected program files on the file server. One program file is corrupted and must be restored from a backup tape.
11. Two days later, the system administrator repeats the process after a user executes an infected program from his local hard disk. This time, the system administrator spends two hours disinfecting the file server, and three hours disinfecting hard disks in workstations.

Detailed Example: 4096

The 4096 [aka 4K or Frodo] virus is a complex program-infecting virus and a prime example of a stealth virus. The virus behaves as described below:

1. The user executes a 4096-infected program.
2. The virus installs itself at the top of memory, then reduces the apparent amount of conventional memory in the system by about 6 KB.
3. The virus hooks interrupt 21H, then turns control over to the infected program.
4. The virus monitors most DOS functions that deal with files. It infects files when DOS functions 3CH (create file), 3DH (open file), or 4BH (exec) are called for EXE or COM files. When 4096 infects a file, it saves the file's original size, time stamp, and starting address in the copy of virus code appended to the file. To mark a file as infected, 4096 increments the year on the file's time stamp by 100 years. DIR only lists the last two digits of the year, so this change will go undetected, even when the virus is not resident in memory.
5. When a program calls the DOS functions to get a file's size or time stamp, 4096 checks to see whether the file is infected, and if so, returns the correct data for the original (uninfected) file.

6. If a program attempts to read the EXE file header from an infected file, 4096 returns the header with the corrected starting address for the program.

7. The virus also intercepts calls to DOS function 30H (Get DOS Version). When this function is called, 4096 checks the date on the system clock. If the date is September 22 or later, the virus hangs the machine. Actually, the system crash is due to a bug in the virus. The virus overwrites the Master Boot Sector on hard disks and the boot sector on floppy disks with a program which displays the message ‘FRODO LIVES!’ in large letters on screen.

NetWare and Viruses

Many viruses simply are not written to coexist with *NetWare*. Most variants of the Jerusalem B virus (one of the most common DOS viruses), when executed, appropriates some of the same interrupt functions which *NetWare* uses. This causes the shell to lose its connection to the default file server as soon as Jerusalem B becomes active, which prevents the virus from infecting other files on the server.

A virus, like all DOS programs, controls the machine through DOS and BIOS function calls. The way in which *NetWare’s* workstation shell isolates the file server from low-level system calls prevents viruses from damaging FAT tables, disk sectors, or other low-level disk structures on a file server. Figure 1. shows the relationship between applications, the shell, DOS, BIOS, and the *NetWare* operating system.

The shell intercepts some of the DOS calls and redirects them over the network to the file server. Program-infecting viruses can infect file through DOS function calls, which might be redirected to the server.

The low-level functions which operate at the disk sector level are BIOS calls which are not redirected. Because many viruses rely on BIOS interrupts for access to disks, many viruses are blocked from taking any action when

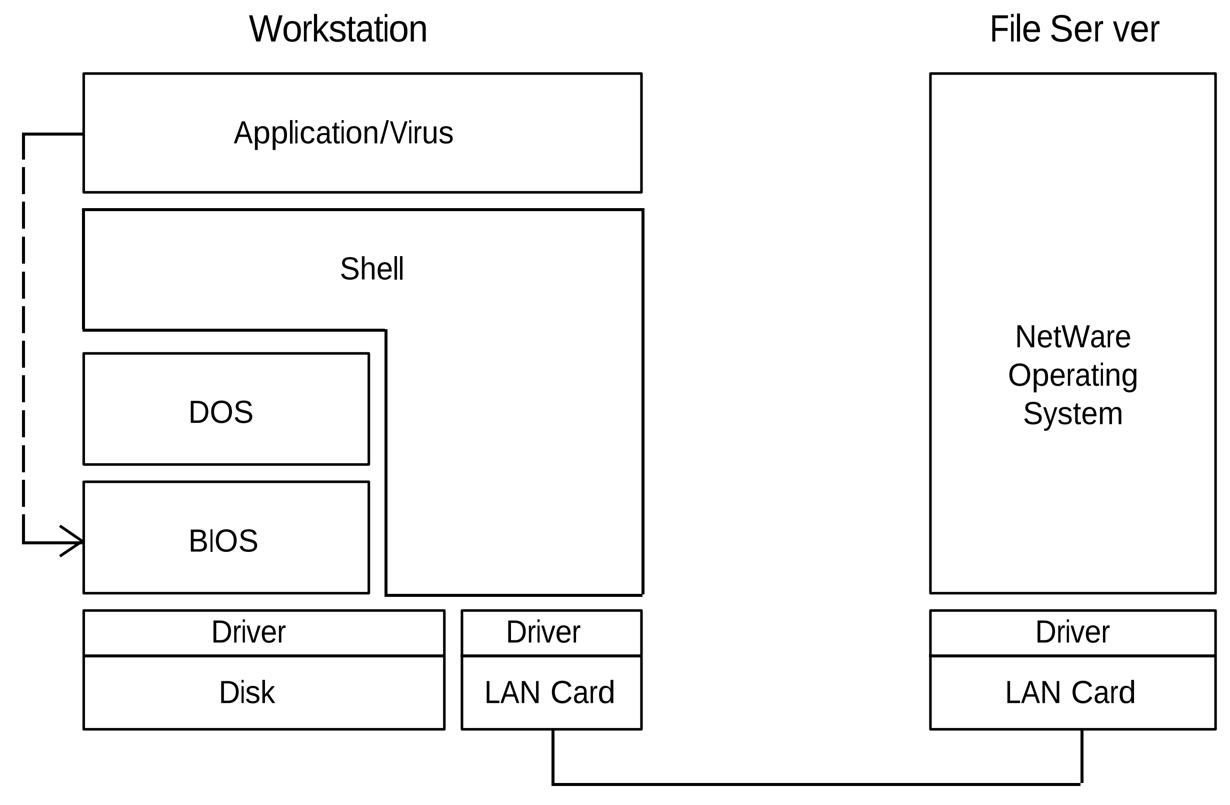


Figure 1. Relationships between workstation and file server software components.

running on a network workstation. Viruses are restricted to using DOS function calls when acting against files on a file server. *NetWare's* security features can limit the amount of damage which a virus can do to a file server.

Viruses take advantage of DOS and BIOS functions, but *NetWare's* security features are separate from those of DOS, and therefore can block virus actions. For example, program files on a *NetWare* file server have an Execute-Only attribute which does not exist under DOS. If the Execute-Only attribute is set, the executable file cannot be modified, even if the user has Read and Write rights in the subdirectory.

Once it is set, the Execute-Only attribute cannot be cleared, nor can the executable file be opened for Read and Write. This means that even if a *NetWare*-aware virus were written, such a virus could not infect a program flagged Execute-Only using the obvious method of copying the file to another name, infecting the new file, deleting the old file, then renaming the new file to the name of the old file.

The Read-Only attribute on files is a feature of both DOS and *NetWare*, and most viruses are written to circumvent it. On a *NetWare* volume, however, a user's right to change the setting of the Read-Only attribute can be restricted by not granting the Access Control right (or Modify right under NetWare v.2.xx) in directories containing executable files.

The Delete Inhibit and Rename Inhibit attributes are other features unique to *NetWare* files that can prevent viruses from infecting programs on a file server. The combination of directory rights and file attributes are sufficient to prevent any DOS virus from infecting any program on a *NetWare* file server.

Remaining Problems

Two serious problems remain. First, a few programs store configuration information in their own executable files. These files cannot be flagged Execute Only, as they cannot then modify themselves. Since the user must have rights to modify the file in order to save the configuration information, the file is vulnerable to virus infection. One solution is to create a separate 'maintainence' account that has Write access to the executable file. When a user wants to change the program configuration, he or she must log in as the maintainence user for the program. The login script for the maintainence account could run a virus scanning program to verify that there is not a virus active in memory before the program is reconfigured.

A second problem is that of data files. Program files can be protected from modification by viruses, but users must have rights to modify data files. If users can modify them, viruses can corrupt them. The only solutions to this problem are the consistent use of virus scanning software on workstations (to ensure that viruses are not introduced to the network) and the frequent backup of data files.

Prescriptions

Here are some measures you can take to protect your network against a virus attack:

☞ Flag .EXE and .COM files as Read-Only and Execute-Only (if possible), and set the Delete Inhibit and Rename Inhibit attributes on the files (unless the program stores configuration information internally).

☞ Grant only Read and File Scan rights to all users in PUBLIC, LOGIN, and application directories.

☞ Do not routinely log in as SUPERVISOR or Supervisor-equivalent.

☞ Back up data files frequently.

☞ Maintain layers of backups.

☞ Prepare a write-protected, bootable disk with virus scan and removal software.

☞ Become familiar with different types of viruses, so you know the appropriate steps to take for whatever virus you might encounter.

☞ Educate your network users regarding network security features and how to use them, symptoms of virus infection, and corporate procedures in dealing with suspected infections.

If you suspect a virus infection, there are certain steps which **must** be followed:

☞ Do **NOT** log in as SUPERVISOR.

☞ Power-off the suspect machine.

☞ Boot with a write-protected DOS disk.

☞ Run virus scan software on local disks.

☞ Delete or disinfect infected files as directed by the virus scan program.

☞ Login as a non-SUPERVISOR user with only Read and File Scan rights to the system.

☞ Scan all visible files, and log the names of infected files.

☞ Login to the server using an account with the MINIMUM rights necessary to treat infected files as directed by the virus scan software.

NetWare-Specific Viruses

To date, only two *NetWare*-specific viruses have been reported. Neither poses a threat to networks, and neither has been found ‘in the wild’ outside research facilities.

The GP1 or Get Password 1 virus was discovered in Europe in 1991. It is a variant of the Jerusalem virus that attempts to gather user passwords. The virus checks for the *NetWare* DOS shell in memory, and if it is present, attempts to capture the user’s password from memory when he or she logs in. The virus then broadcasts the password over the network to a specific socket number. A companion program must be running somewhere on the network to gather the passwords as they are broadcast.

GP1 does not work with versions of *NetWare* released since 1987, because since that time, the shell has encrypted passwords before sending them.

The second *NetWare*-specific virus, CZ2986, was discovered in Czechoslovakia in 1991. The virus places itself resident in memory and intercepts calls to the *NetWare* function that logs a workstation in to file servers. The virus collects 15 username/password combinations and saves them in the infected file. A separate program would have to inspect the contents of this file to gather the usernames and passwords. This virus can be easily defeated by setting directory rights and file attributes.

NetWare-specific viruses do not exist in great numbers for at least three reasons:

➤ *NetWare* is not yet as ubiquitous as DOS.

➤ *NetWare* programming knowledge is not as widely disseminated as DOS programming knowledge.

➤ *NetWare’s* security features are much more comprehensive than those of DOS (which consists solely of the Read-Only file attribute which a virus can clear).

NLM-Based Viruses

In assessing the virus threat, one must also consider the possibility of server-based viruses. *NetWare* 3.x server based processes (*NetWare Loadable Modules*) are written in ANSI C and compiled into 32-bit Intel ’386 code. Is it possible for a virus to infect these processes?

NetWare Loadable Modules have four forms. LAN drivers (.LAN), disk drivers (.DSK), name spaces (.NAM) and applications (.NLM). All execute ’386 code. However, the file formats of these executables differ from DOS executables, so infection by DOS viruses is impossible. In order to develop an NLM virus, an author would have to gain understanding of the low level structure of loadable modules. Furthermore, these files are stored in the SYSTEM directory, which has Write-access available only to the supervisor.

Protection from a classical virus implemented as an NLM is possible by using ‘memory bounds check’ software such as *NuMega’s Net Check* NLM. This reconfigures the ’386 server CPU, placing the server operating system in a protected memory environment (‘ring 3’). Any NLMs which attempt to reconfigure themselves or make illegal memory accesses will generate hardware interrupts, activating *Net Check* which suspends the offending NLM.

Unauthorised NLMs

Although the possibility of viruses infecting NLMs is remote, a more realistic threat exists in the form of unauthorised NLMs. The file server environment is a privileged one - NLMs running in the server have security privileges equivalent to the Supervisor. Therfore, if access can be made to the file server console, then it is possible to load unauthorised and potentially damaging NLMs.

Two strategies can be adopted to deal with this threat. Firstly, restrict physical access to the server. If a server can be accessed freely, then data integrity cannot be assured, no matter what level of security is employed. The Server can be unplugged, knocked (causing hard disk damage) or otherwise interfered with. Ideally, servers should be placed in dedicated areas with air and power conditioning.

Secondly, use the SECURE CONSOLE console command. This restricts NLMs from being loaded from anywhere other than the SYS:SYSTEM directory on the server. Furthermore, SECURE CONSOLE unloads the DOS command processor and operating system from server memory (*NetWare* 3.x servers use DOS to bootstrap the server operating system - the DOS code is then only used to access the local DOS drive for loading NLMs). Once the DOS code is unloaded, the local DOS drives cannot be used to load NLMs. An additional benefit is that the memory formerly occupied by DOS is given to the server file cache.

However, another entry point for unauthorised NLMs exists: the remote console, RCONSOLE. This is a management utility that facilitates remote control of the file server console from any network-attached DOS workstation, provided the user knows either the Supervisor password or the RCONSOLE administration password. One of the functions of RCONSOLE is uploading server-based software from the remote workstation to the remote server. If an intruder has access to an attached workstation, RCONSOLE and either the Supervisor password or the RCONSOLE maintainence password, then he could upload an unauthorised NLM.

To protect against this possibility, always ensure that the RCONSOLE and supervisor passwords are well guarded, or if possible, do not enable the optional RCONSOLE maintenance password. If these guidelines are met, the threat from RCONSOLE recedes to that of unauthorised access of the supervisor account.

If, in the worst case, an unauthorised NLM gains control of a *NetWare* server, potential damage is limited to the machine that the NLM is running on. Although NLMs can log on to another server, they are mediated by the security of that remote server. Without a username and password, an NLM cannot gain access. NLM security under these conditions is equivalent to the case of trying to gain user access without username and password.

Server-Based Virus Scanning Software

In the *NetWare* environment, virus scanning software can be implemented as *NetWare Loadable Modules*. Products such as *Lanprotect* (from *Intel Corporation*) monitor viruses at the server and are thus immune from interference by stealth viruses. Typically, these programs scan .COM, .EXE, .OVL, .NLM, .DSK, .LAN and .NAM files, generating CRCs. These scans can be server-intensive and are thus usually scheduled to run at times of low load.

Limitations of NLM Virus Scanners

A current limitation of server-based virus protection programs is that they tend to scan for only *Intel* 80x86 executable viruses. *NetWare* servers often support workstations based on other processor designs, e.g Macintosh, and many *SPARC* based Unix workstations. Viruses designed for these environments will not be detected by these scanners. To protect these machines, client-based scanning facilities should be used.

Security Audit NLM

In the event of a breach of LAN security by a virus, unauthorised NLM or an intruder, it is important to be able to re-establish system integrity. To do this, all important system-critical files need to be examined for evidence of change. This is the function of system auditing. *NetWare* 3.x can be enabled for system auditing by running a third-party audit NLM, which expands *NetWare* base services to include auditing.

This *NetWare Loadable Module* (NLM) for *NetWare* v3.11 servers provides a number of additional audit functions, including login audit, bindery and trustee audit, and file audit. File audit can be especially useful when applied to the file types mentioned above - any record of updates of NLMs, or DOS executables could be traced back to specific machines, thereby identifying the source of the virus.

Questions and Comments

Security against viruses is an area of ongoing research for *Novell Systems Research*, and will be addressed in future *AppNotes*. Please direct questions and comments to:

Morgan Adair, Technical Consultant, *Novell, Inc*.
Mail Stop E-23-1, 122 East 1700 South, Provo, Utah 84606, USA.
Tel: (801) 429-7757, Fax: (801) 429-5511
Compuserve Mail: 76424,410
MHS: MADAIR@NOVELL
Internet: madair@novell.com

The full paper is available in its original form as a *Novell AppNote* (July 1992). Reprints of *AppNotes* can be obtained at a cost of US$15 each. Subscription to *AppNotes* is available at a yearly cost of US$95. Call *Novell Research*, in the US, on 801 429 5380.

Thanks to Eric Babcock of *Novell Security*, who provided a secure lab and virus samples used in research for the original *AppNote*. Thanks also to Steve Chang, President of *Trend Micro Devices Inc*., who provided technical data on a number of viruses and a copy of *Trend's PC-cillin*.

VIRUS ANALYSIS 1

Jim Bates

V-SIGN - A Supplementary Report

The increasing prevalence of the V-SIGN virus warrants a supplementary report to the analysis of this virus which appeared in *Virus Bulletin*, July 1992, page 6.

Unusual Features

The virus is known as V-SIGN due to its display when the trigger routine executes. V-SIGN is a boot sector virus which infects floppy disks and the Master Boot Sector of hard disks. It hooks itself into the system at boot-time in the usual way via INT 13H and it attempts to detect and remove the New Zealand (Stoned) virus. In this respect it may have some claim to being an anti-virus virus but the implementation is so convoluted and so poorly executed that it actually increases the chance of system malfunction.

The infection method differs from most boot sector viruses in that no attempt is made to store a clean copy of the original boot sector. The integrity of the bootstrap process is maintained by swapping chunks of code around.

Another new development in a boot sector virus is the use of code randomisation to make pattern recognition difficult.

Installation

During infection the virus collects 38 bytes of code from the Master Boot Sector of the target disk, stores them within itself and replaces them with code that redirects the boot process to load the remainder of the virus from elsewhere on the disk. The virus itself is actually only 786 bytes long but this hook which is inserted into the original Master Boot Sector could be said to extend its total length to 824 bytes.

Once the virus is in memory it checks whether it is already installed. This might seem superfluous but with the increase in alternative and selective boot software, it is quite possible for a machine to initiate a warm boot sequence where existing system hooks have not been reset. The method of self-recognition is to collect the segment portion of the INT 13H vector and check that segment of memory for a recognition word of 9876H at offset 0D6H.

An ‘Anti-Virus Virus’

If this is not found, a check is then attempted to detect the existence of the New Zealand virus. The intention is to compare a 10 byte fragment of code found in the New Zealand virus with a specific memory location. The code fragment is obviously there simply for pattern matching since it is never executed. However, this check fails due to an error in the program and V-SIGN will not detect New Zealand at this stage. Because this check fails, the code to unhook the New Zealand virus is never executed and processing passes immediately to the installation routine.

Installation is achieved by the familiar method of locating the top of conventional memory and moving the code up into it. Then the memory pointers are modified so that subsequent programs will not use that space and finally any relevant interrupt handling routines are hooked into the system interrupt table. The virus then repairs the displaced boot code in memory and finally passes control to it so that the normal boot routine may continue.

Operation

V-SIGN only intercepts the BIOS INT 13H system function. During these system requests, the interception routine examines the status of the floppy drive motors. If neither the A or B drives is running, the routine aborts. Otherwise, checks are made to ensure that the request is either to READ or WRITE to the disk in the first two floppy drives or the first two hard drives. Once these parameters are established, the routine decides whether the request is for hard or floppy disk and branches accordingly. If the request is for access to a hard disk, the routine sets a flag and exits. Floppy access, on the other hand, results in the virus checking the disk to determine its layout and whether it is already infected. If the disk is suitable for infection, the trigger counter is incremented and tested for divisibility by 63. If the condition is not met, processing continues with the code randomisation mentioned above.

This consist of swapping the position of certain instructions within the code to be inserted into the Master Boot Sector. This is undoubtedly done to confuse scanning software and make recognition difficult. The effect is to produce six slightly different versions of the virus on a cyclical basis.

Once this modification is complete, a second check is made for the New Zealand virus. This time the test is done properly (using the recognition pattern again) on the existing Master Boot Sector and if New Zealand is found, it is removed and replaced by the original Master Boot Sector from the relevant sector of the boot track (Track 0, Head 0, Sector 7 for hard disks, Sector 3 side 1 for 360K floppies).

The Wrong Bit

At this point, a check is also made to discover whether the target disk was previously infected with V-SIGN. It is here that an interesting error occurs - the code which is supposed to recognise the existence of V-SIGN on the disk compares a specific location within itself with a similar location on disk. However, one of the addresses is wrong in such a way that the check will fail. V-SIGN will thus re-infect disks and in so doing will irreparably damage the boot sequence.

Of course this will not affect floppy disks used only for data, and will only occur on hard disks if the Master Boot Sector is read once the virus is resident. The error is caused by a single incorrect bit in the code. The nature of the mistake makes it unlikely that it was done deliberately. However, it could conceivably be due to 'degeneration' (see *Technical Notes*, page 4). It should be noted that this error turns a 'benign' virus into a destructive one.

During the infection process, the virus calculates exactly where to place its code depending upon the format of the disk in accordance with the following table:

Target Disk	Track	Head	Sectors
Hard disk	0	0	4/5
5.25" 360k	0	1	2/3
5.25" 1.2M	0	1	13/14
3.5" 720k	0	1	4/5
3.5" 1.44M	0	1*	14/15

* Error - should be Head 0.

Apart from the hard disk and the last floppy, these represent the last two sectors allocated for use as directory sectors. This will cause corruption and data loss on disks with files approaching the maximum number within the root directory. The 3.5" 1.44M floppy is an obvious mistake since it results in the virus code being stored in the first data sectors on the disk and these will be the first to contain data!

Trigger Routine

As mentioned above, this occurs every time the counter passes an exact multiple of 63. The display is a simple one of a red 'V' shown on a black background. The 'V' is made up of block characters on a text screen and will therefore display properly even on old monochrome monitors. Once the display is completed, the computer will go into an infinite loop and hang. This will certainly cause data corruption if it occurs during file update procedures.

Disinfection

Disinfection must be undertaken in a clean DOS environment, i.e. having booted from a clean system disk.

On hard disks the 38 bytes of virus pointer code embedded in the Master Boot Sector (Track 0, Head 0, Sector 1) start at offset 36H. The actual pointer to the sector number where the remainder of the virus code is stored will be found at 4EH (on hard disks this will be at Track 0, Sector 4 which is usually unused). The original 38 bytes of good boot sector code in sector 4 are located at offset 6H, the first byte of which will be EBH which is the very first byte of the Master Boot Sector. The 38 bytes starting at offset 8H are the displaced boot code and should be written to offset 36H in the Master Boot Sector.

NOTE: If a backup copy of the hard disk boot sector is available, none of this convoluted procedure is necessary. The PC can be restored with the minimum of fuss. Under DOS 5 the virus can be removed from the Master Boot Sector using the straightforward FDISK/MBR syntax.

Files can safely be transferred from diskettes using the DOS COPY command and infected diskettes should then be formatted under clean system conditions.

V-SIGN

Virus name :	V-SIGN
Aliases :	Cansu
Type :	Resident boot sector virus.
Infection :	Infects logical sector 0 of diskettes (any density) using the last two sectors of the root directory to store the remainder of its code. On hard disks the virus inserts 38 bytes of pointer code into the Master Boot Sector (Track 0, Head 0, Sector 1). The remainder of the virus is stored at Sectors 4 and 5.
Recognition :	
Disk	Value of 9876H at offset 5AH in the Master Boot Sector.

Hex Pattern

```
1372 FA?? ???? ???? ??B9 0E00 BA00 01CD 1372
EAE9 A601 7698
```

System	9876H value at offset 0D6H into segment pointed to by segment portion of INT 13H vector.
Intercepts :	INT 13H for infection and calculation of trigger conditions.
Trigger :	Displays large 'V' sign in block characters on a text screen. Sign will be red on black on colour monitors.

VIRUS ANALYSIS 2

W-13a - The 'Toothless' Virus Starts to Byte

Another virus originally classified as rare, endangered or extinct has now been reported at large in the UK. This is the W-13a virus which was originally reported from Poland in December 1989. The virus also became known as 'Toothless' because it has no trigger routine, but this name has since fallen into disuse. The main code is based on the Vienna virus but with altered search routines. There are two major versions: W-13a and W-13b, the differences are minor and the essential attributes remain the same.

Operation

There is no installation routine since this is not a resident virus; it executes when the host file is loaded and then passes control to the host. Processing begins by checking that the operating system is later than DOS 1. This is to ensure that the system services which the virus will use are correctly supported. The next routine sets up a temporary Disk Transfer Area (DTA) and begins searching for COM files in the current directory. As each COM file is found, it is checked to see whether it is already infected. If so the search continues.

The routine then reverts to the parent of the current directory and continues the search there. However, the coding is untidy and does not work, so once the current directory has been infected, no further infections will occur. It should be made clear that once a clean file is found, it is infected and the virus then passes control to the host. This means that only one new infection can be generated at each pass of the virus.

The virus uses a recognition signature similar to the ubiquitous 62 second marker - in this case the months field is set to 13 when a file is infected. Only COM files are infected (by reference to the file extension only) and the code is appended in the usual way. Target files must have an uninfected length of between 256 and 64000 bytes (inclusive) to be suitable for infection. The length of the appended code is 534 bytes with W-13a and 507 bytes with W-13b although in both cases the actual length of the virus code is less than this (377 and 356 respectively).

The library specimens which *VB* has are both originals - that is they are straight from the author and not copies derived through replication. Within the files are some comments in Polish, assembler source code (not for the virus) and such words as WABIK (the Polish word for 'decoy') and RYBKAI. There are several bugs in the code, the most serious of which prevents the repair of target file attributes (e.g.: to clear a READ ONLY). The original date and time (except for the month 13) of infected files are restored. It is also worth noting that DOS reports the month as 13 without registering an error.

Conclusions

Another seemingly extinct virus makes an appearance in its original form. This may be a deliberate re-infection or it may be due to over-zealous (and under-cautious) researchers mishandling such code. However it happened, the result is yet another virus spreading itself in the wild. Fortunately, recognition is easy and disinfection is straightforward.

W-13

Virus :	W-13a and W-13b
Aliases :	Toothless
Type :	Non-resident Parasitic virus
Infection :	COM files between 257 and 63999 bytes long (inclusive)
Infective Length :	W-13a = 534 bytes (appended), W-13b = 507 bytes (appended)

Recognition :

File Value of 13 in the month field of the file date/ time stamp.

Hex Pattern

```
8BD7 2BF9 83C7 0205 0301 03C1 8905 B440 8BFA
2BD1 B9?? ??CD
```

Note : the above hexadecimal pattern is augmented to 24 bytes from that given in *VB*, January 1991.

System	No system recognition
Intercepts :	No intercepts
Trigger :	No trigger
Removal :	Specific and generic disinfection is possible. Under clean system conditions, identify and replace infected files.

Researchers should note that the two wild bytes (denoted by question marks in the hex pattern) represent the infective length of the virus and have the following values in the known versions of the virus:

W-13a	16 02	signature is at offset 250 into the virus code
W-13b	FB 01	signature is at offset 224 into the virus code

VIRUS ANALYSIS 3

Keypress

The Keypress virus, first reported in the United States in October 1990, has recently been reported from one or two sites in the UK. It can cause confusion as its trigger routine simulates a faulty keyboard. While the virus is primitive, the code shows signs of having been written by someone with experience in assembler programming. It is also the only virus anlaysed so far which invokes different routines depending upon the DOS version on the host machine.

Installation

Keypress is a resident, parasitic virus which infects all executable files under DOS 3.00 and above but which infects COM and EXE files under earlier DOS releases.

The virus appends itself to the end of the host file and is executed immediately the file is run. The code calls a register save routine and checks its own host to determine whether or not it is a segmented EXE file. Depending on the result of the check, the altered sections of the host file are repaired and the virus then issues an 'RU There?' call by examining offset 600H in segment 0 for a value of 1. If found, this indicates that the virus is resident and the code returns to the host program.

If the virus is not located, an installation routine is called which checks the chain of Memory Control Blocks and modifies the last one to enable installation of the virus code. Once the code has been physically relocated into the newly created MCB, a final routine is called which hooks the resident code into the system services. After this, processing returns to the original host program leaving itself resident and active in memory. This division of the virus into discrete subroutines is done in an assured manner indicating that the writer is no novice.

Operation

The virus intercepts two of the interrupt service routines, INT 1CH and INT 21H. The INT 21H service routine handles all DOS requests. During installation, a check is made of the DOS version number operating on the host machine. One interception routine is hooked for version 3 or later, while a different one applies to earlier versions.

Under DOS 3.xx and higher releases of the operating system the virus intercepts requests to LOAD and EXECUTE program files. Each file passed for execution is examined by the virus to determine whether it is already infected. For files with the 'MZ' EXE header, this entails testing for a value of 133H in the start offset portion of the header (the IP Field). Non-EXE files are checked by comparing 12 bytes of their code (starting at the fifth byte of the file) with a matching section within the virus.

Any size of EXE file is considered suitable for infection, but non-EXE files must have an uninfected length of between 1216 and 64063 bytes (inclusive). It is worth noting that EXE type files which have a value of zero in the MAXALLOC field of the header are deliberately excluded from infection.

Earlier DOS Versions

Under earlier versions of DOS the virus intercepts calls to OPEN files for READ ONLY access. The target file is checked to see whether it has a COM or EXE extension and if so, processing continues with the series of infection routines and conditions used for the other interception. Files with any other extension are not infected. It is quite possible for COMMAND.COM to become infected under either version but the infection conditions make this less likely.

During interceptions of either type, temporary service routines to INT 23H and INT 24H are installed to handle potential errors. The virus attempts to write to target files without checking whether the READ ONLY attribute is set. During infection there is also no attempt to preserve the original date and time stamp of the file, nor is there any attempt to hide the subsequent increase in file length. The virus writer obviously had problems with segment arithmetic and chose an odd way to ensure correct offset location of his code when infecting EXE files. This results in quite a wide variation in infective lengths (see below).

Trigger Routine

The INT 1CH service is referred to as the "timer tick" routine. This is executed during normal operation at a rate of approximately 18 times per second. The virus interception routine increments a counter during each pass and checks for values of 10800 and 10836 respectively.

While the counter is between these two values, a routine generates extra keyboard interrupts. The effect is to repeat keystrokes up to five times; thus if the user types 'abc', the keyboard will output 'aaaaabbbbbccccc'. Since the counter is set to zero when the virus is first installed, it will take approximately ten minutes to reach the first value (10800/18 = 600 seconds) and a further two seconds to reach the next. Once it passes 10836, it is zeroed and the process begins again - i.e. two seconds of unpredictable keyboard activity every ten minutes.

Conclusion

The simulation of a keyboard fault will prove an annoying inconvenience to users of afflicted PCs. Fortunately, the virus is not destructive, is easy to detect and should be removed by the simple expedient of deleting infected files and replacing them with known clean copies. As always, the system should be booted from a clean write-protected system diskette before disinfection proceeds.

Keypress

Virus :	Keypress
Aliases :	Turku, Twins
Type :	Memory-resident Parasitic file infector - COM, EXE and executables (including COMMAND.COM)
Infection :	COM files between 1216 and 64063 bytes long (inclusive) EXE and other executables of any length.
	Adds from 1472 to 1487 bytes to EXE files
	Adds from 1216 to 1231 bytes to other files.
Recognition :	
File	0133H in IP field of EXE file headers. Pattern recognition in other types.
Hex Pattern :	

```
7405 C707 0100 F9F5 1FC3 F606 1801 0174 0D8C
C005 1000 0106
```

System	0001H value in 0000:0600H indicates virus is resident.
Intercepts :	INT 21H for infection.
	INT 1CH for trigger counter
	INT 24H for temporary internal error handling
	INT 23H temporarily to prevent control -break interruption.
Trigger :	Generates keyboard errors for two seconds every ten minutes.
Removal :	Specific and generic disinfection is possible. Under clean system conditions, identify and replace infected files.

Note that in keeping with the increased length of the recognition details, the above hex recognition pattern is augmented to 24 bytes from that given in *VB* January 1991.

PRODUCT REVIEW

Mark Hamilton

Disknet

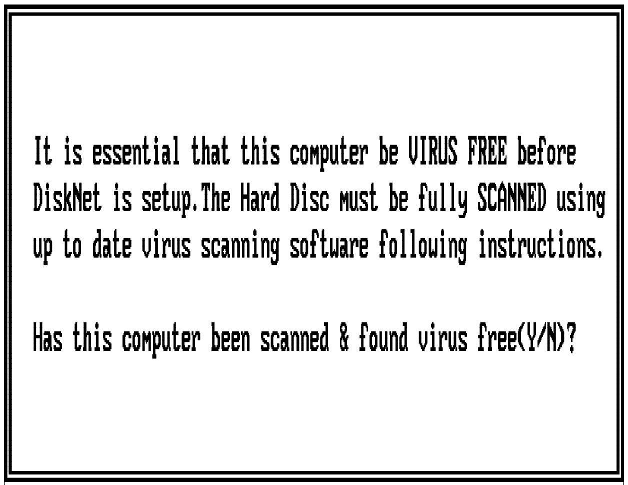
One of the great problems in any corporate environment is keeping virus-specific anti-virus software up-to-date. If users access this software over a local area network, it's simply a case of ensuring that the software on the various servers is regularly updated. For users of standalone PCs, however, either the users must be trusted to update their own defences or the MIS department must undertake the checking for them - often outside regular working hours. Sound practice, repeating this exercise each and every month, is a time-consuming task.

User compliance - ensuring that users regularly run this software - is also difficult to enforce. How many users run their scanners and integrity checkers regularly? Equally worrying are issues of software piracy and theft; the prospect of *FAST* in the UK or the *SPA* in the United States conducting an on-site investigation is most unwelcome to most beleaguered MIS departments.

A couple of months ago, I looked at *D-FENCE* from *Sophos* and its particular solution to the various problems listed above. Another British company, *Reflex Magnetics*, a specialist software duplication and copy-protection house in London, offers a somewhat similar product called *Disknet*.

The theory behind *Disknet* is disarmingly simple: you allocate one or more PCs, under the control of company 'trustees', and set them up as 'Gateways'. The remaining PCs have *Disknet* permanently loaded in memory. All disks have to be converted by a Gateway PC before they can be used on one of the protected ones. The user is then prevented from using any disks which have not been checked and marked with a signature.

Authorisation packages such as *Disknet* impose certain inherent restrictions. One subtle consideration, worth pointing out, arises when users wish to install brand new software. In such instances the Gateway PC 'trustee' should copy the the master disks and then validate the copies - irrespective of whether the software is delivered on write-protected media or not. This is because most software houses only warrant the *physical media* against defect and will take a jaundiced view if master disks have to be returned, for any reason, and have been 'tampered' with by an authorisation product such as *Disknet*. [Copying master disks is not *strictly* necessary. The Supervisor's menu can be used to disable and re-enable *Disknet*. Ed.]



```
It is essential that this computer be VIRUS FREE before
DiskNet is setup.The Hard Disc must be fully SCANNED using
up to date virus scanning software following instructions.

Has this computer been scanned & found virus free(Y/N)?
```

Disknet currently supports nine of the best known virus scanning products and clearly instructs the user that scanning must be undertaken prior to installation.

Manual

The twenty page manual is terse but provides all the information necessary to install and use the software. I was amused to notice that the index lists separately 'Hard Disc', 'HARD DISK', 'Hard Disk', 'Hard disk' and 'hard disk' - there are a number of similarly duplicated entries.

Installation

Disknet is delivered on both 5.25 and 3.5-inch media and the disks contain 16 files, nine of which are shell-type programs used to invoke the user's chosen anti-virus scanner. The following products are supported:

- *McAfee Associates' Scan*
- *S&S International's Findvirus*
- *Frisk Software's F-Prot*
- *Norton Anti-Virus version 2.x*
- *Bates' VIS version 3.x*
- *Sophos' Sweep*
- *Visionsoft's SmartScan*
- *Central Point Anti-Virus*
- *Thecia Software's Vclean* [What is this? Ed.]

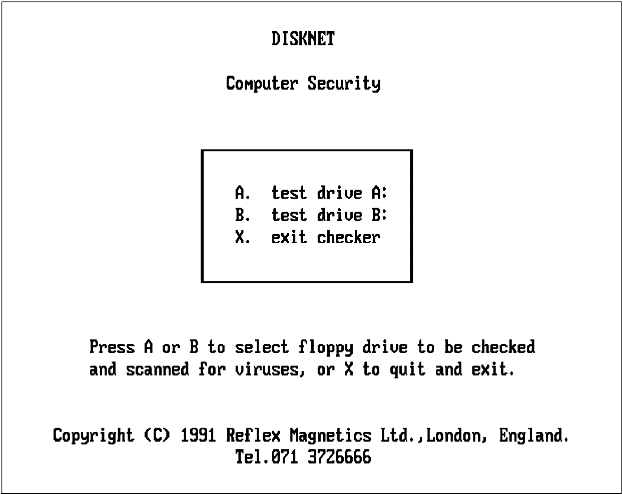
The software must first be installed on a Gateway PC and a DOS system diskette (without write-protection at this stage) prepared. The installation program should then be run. One of the very first things it asks is whether or not the hard drive has been scanned for viruses. If this has not been done, the installation program aborts and instructs that such scanning should be undertaken. You are then asked whether you wish to install its own access control program (*CURE*) or another access control package of your choosing. The program makes some changes to the AUTOEXEC.BAT file (which are not revealed) and the original file is saved as AUTOEXEC.OLD. The system disk should then be inserted into drive A. Install places four files on this floppy and creates an AUTOEXEC.BAT file. The system diskette should be write-protected at this stage. The purpose of this floppy is to check the hard drive and also to boot the PC without *Disknet's* protection so that you can remove or disable it.

Having set up the Gateway PC, the software should be installed on all the other PCs in the company. Each PC must first be swept for known viruses using a scanner prior to installing *Disknet*. The installation program runs and requests that you insert a newly created system diskette onto which it writes all relevant hard disk boot sectors. If a boot sector virus is subsequently detected by the *CURE* program (which is automatically run when the machine is booted), it is replaced using a saved copy taken at installation time and stored on the boot floppy.

In Use

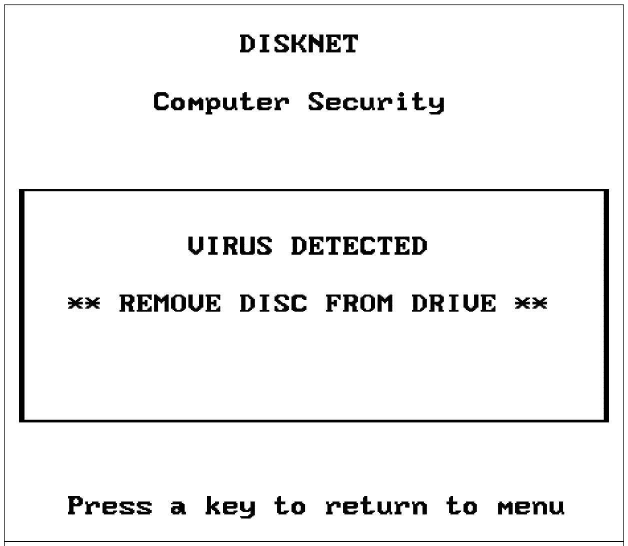
The only difference between a Gateway PC and a non-Gateway PC is that the Gateway PCs have a copy of *Disknet's* scanner shell as well as the scanning software.

Whenever a floppy disk is accessed, *Disknet* checks to see whether its hidden signature appears on the disk and if its contents agree with the disk's contents. If either are false, it pops a message on the screen: 'UNVALIDATED DISK: please have this checked' and forces the DOS 'General failure reading drive x, Abort, Retry, Fail?' message. That's



```
                    DISKNET

               Computer Security

              A.  test drive A:
              B.  test drive B:
              X.  exit checker

   Press A or B to select floppy drive to be checked
   and scanned for viruses, or X to quit and exit.

 Copyright (C) 1991 Reflex Magnetics Ltd.,London, England.
                   Tel.071 3726666
```

When *Disknet* is installed on company PCs it enforces disk validation and virus screening. A simple scanning menu provides the options to check floppy drives.



```
DISKNET

Computer Security

VIRUS DETECTED

** REMOVE DISC FROM DRIVE **

Press a key to return to menu
```

The initial integrity of the machines protected by *Disknet* is vital and the use of the very latest scanning software is recommended during the authorisation process.

fine under the DOS command interpreter (i.e. at the DOS prompt) or within a DOS application. However, under *Microsoft Windows* 3.1, any unauthorised diskette is rebuffed and a message box informs you that the disk has not been formatted (i.e. DOS formatted), and asks whether the user wishes to format it. I would suggest that this is highly misleading because the error is due to the diskette being rejected by *Disknet* and not because it is unformatted; this could result in users accidentally reformatting diskettes which contain valuable information. It would be preferable if *Disknet* were to return an alternative error code to *Windows* 3.1 - one that caused a less dangerous error message to be displayed.

Footprint

Disknet occupies 3,920 bytes in memory - not an onerous amount - and it didn't interfere with any memory-resident programs it was tested alongside (*Borland's Sidekick*, various mouse drivers, CD ROM drivers, memory managers, *Total Control's VISMON*). Under DOS 5, DR-DOS 5 or 6, or any version of DOS run in conjunction with a third-party memory manager which provides upper memory blocks, *Disknet* can be loaded 'high' so that there's no conventional memory footprint.

If an unauthorised diskette is used in a PC without *Disknet* installed, and its contents changed (i.e. it is written to), the unique authorisation on the disk is overwritten, thus forcing the diskette to be re-scanned and a new signature applied at the Gateway PC. The Acid Test is thus whether *Disknet* can detect viral activity that has occurred on an authorised disk. It appears that *Disknet* checksums the root directory, the File Allocation Table, date and time stamps and the boot sectors of diskettes - all those critical areas which any functioning program or virus would be expected to change.

Prior to conducting the authorisation process, I thus ignored *Disknet's* imperative to scan for viruses (an order, it must be said, which a real user would find virtually *impossible* to overlook). I subsequently infected a file on an authorised diskette using a non-*Disknet* PC with the Jerusalem (parasitic) virus - the diskette was immediately rejected by the *Disknet* PC as was an authorised diskette infected with Michelangelo, itself a boot sector virus.

Devious Experiments

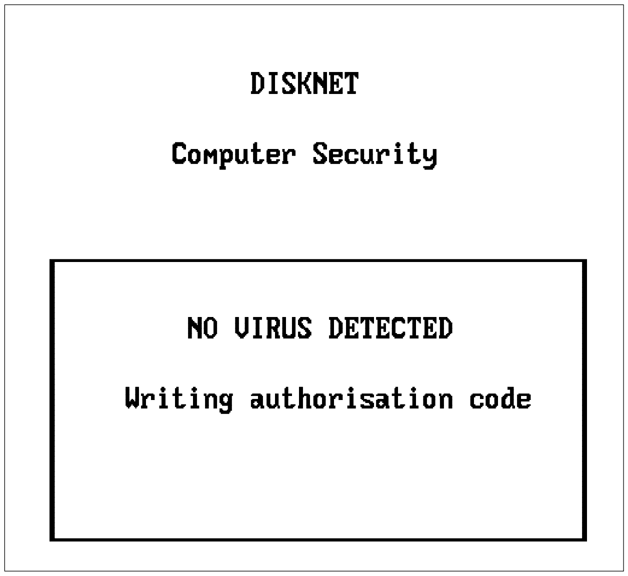
I prepared a floppy with six .COM files, one of which was infected with a parasitic virus. I compressed that file and then scanned the disk - the virus went undetected. I placed the disk in a PC equipped with *Disknet* and it was rejected. So far so good, the disk had not yet been authorised through the Gateway PC. The Gateway PC also pronounced the disk clean and wrote its authorisation signature onto the disk. The disk was then replaced in the *Disknet*-equipped PC which this time accepted it without a grumble. In fairness, I had expected *Disknet* to authorise the diskette with this infected file; so far all that I had proved was that the initial authorisation is heavily dependent on the accuracy of the virus scanning software in use.

I then decided to try a somewhat devious experiment. Suppose you have a program infected by an overwriting virus - if that program is executed, will *Disknet* always detect the alterations introduced by such a virus on an authorised diskette? The answer is no, and this is how I demonstrated it to myself.

I infected a clean COM file with the Burger-405 (a primitive overwriting virus) on an *authorised* diskette but on a *non-Disknet* machine. This infected file was offered to the *Disknet* PC once more which accepted it without complaint.

The Burger-405 virus, which cripples its host program, is a clumsy specimen which poses no real threat whatsoever. The reason, I suspect, that it went undetected, is that it neither changes the length of its host file nor does it alter date and time stamps or even the directory entry.

This may not be an entirely fair test but it does point out that security systems are only secure to a certain extent. In light of this finding, it should be pointed out that when calculating its authorisation signature, *Disknet* does not checksum the entire contents of files and that corruption of files will not necessarily be detected. In more practical and realistic tests, *Disknet* did object whenever a new program



DISKNET

Computer Security

NO VIRUS DETECTED

Writing authorisation code

was added to a previously registered disk or when a program was modified such that its directory information became invalid. In this respect the program works *exactly* as its developer intended.

What Disknet Does

The *Disknet* Gateway PC contains a shell program, CheckXXX (where XXX is a two or three letter code denoting which scanner is being used). CheckXXX invokes the scanner and if the disk is clear, it authorises it by writing a six-byte signature into the floppy's boot sector from offset 4H. From inspection of floppies authorised on the test Gateway PC, this marker always consisted of 'D2XX09' where XX varied according to the disk's content, but I believe that the first and last pair of characters vary from installation to installation.

On a *Disknet*-equipped PC, the signature is kept up-to-date by the resident program. Non-*Disknet*-equipped PCs are not aware of this signature, therefore if a disk's contents are modified outside the *Disknet* environment, *Disknet* complains that the disk is invalid the next time it sees it.

Other Components

There are two optional components which can be installed, C:CURE and LOCK. C:CURE is designed to prevent access to the hard disk if the PC is booted from a floppy diskette while LOCK ensures that *Disknet* is run each time the PC is booted (it removes the reliance on the *Disknet* program being loaded via AUTOEXEC) and prevents unauthorised removal. Neither of these components were fully tested since they make changes to the disk structure and can only be removed using a password provided by *Reflex Magnetics* which is installation dependent.

Disknet And D-FENCE - Relative Merits

Unlike *Sophos' D-FENCE*, *Disknet* from *Reflex Magnetics* does not change the structure of authorised disks - the presence of the authorisation signature does not conflict with the normal operation of that diskette's boot sector - a previously bootable diskette remains bootable after authorisation. This means that users can still take diskettes home, copy files from authorised diskettes onto their own hard disks and work on non-*Disknet* protected machines. *D-FENCE* users can also be given this flexibility should it be required: by default, however, *D-FENCE* disks are unreadable except on *D-FENCE* PCs. In this way, both *D-FENCE* and *Disknet* provide different, but equally valid options; *D-FENCE* forbidding company work on non-company PCs, *Disknet* inherently offering this flexibility.

Summary

Disknet is a well-engineered, rugged program. Under normal working conditions, it performs exactly as it should and, apart from an initial delay in checking diskettes new to the PC, it imposes no discernable operating overhead on reading disks. (Writing to disks typically takes between 20% to 50% longer than without *Disknet* installed.) I have slight reservations about the checksum method used to 'stamp' or authorise diskettes, as corrupted files may go undiagnosed. That said, the method employed will certainly diagnose realistic virus threats.

One final point about authorisation programs, acknowledged by the manufacturers of both *D-FENCE* and *Disknet*. These programs are reliant on the initial integrity of the PCs and diskettes which they protect - the use of the latest scanning software is thus advisable when conducting initial authorisation. With regard to the issue of contamination by unknown viruses, *Disknet* has a slight advantage over *D-FENCE* - any file infections on diskette (subsequent to the initial authorisation) will result in the signature on that disk changing and its consequent rejection by *Disknet*.

Technical Details

Product: *Disknet*

Version: 1.17

Distributor: *Reflex Magnetics Limited*, Unit 1, 31/33 Priory Park Road, Kilburn, London NW6 7UP. Tel 071 372 6666, Fax 071 372 2507.

Availability: IBM PC/XT/AT/PS2 or compatibles running DOS 3.1 or higher. *Windows* and network compatible.

Price: Site licence prices upon request.

Test Platforms: Testing was performed on a GoldStar Goldnote 386SX (the *Disknet* PC), a SIR 486SX (the Gateway PC) and an Apricot Qi-486. All PCs were operating under MS-DOS version 5.

END-NOTES & NEWS

***Barclays Bank Ltd* is estimated to have spent over £250,000** during the last four years on virus suppression according to a report in *Computer Weekly*. Brian Jacques, the bank's security officer, said that the organisation had suffered from around 90 virus incidents and it was encountering viruses at a rate of about 2 per month. Jacques, speaking at annual *DECUS* meeting of DEC users, said that the average cost for cleaning an infection was £3,000. *Barclays Group Information Security* has developed its own anti-virus software (*DEDS* or *Disk Error Detection System*), the use of which is mandatory. The company has 20,000 PCs worldwide. Paul Faulkner of *Barclays Bank Group Information Security* described the bank's approach to virus control at the *2nd International Virus Bulletin Conference*, and a report will be published in October.

Sophos UK has released its *SWEEP* virus scanner as a Network Loadable Module for *Novell NetWare* file servers. *Sophos* claims that the software, which can be run as a permanent background process, is immune from the hiding mechanisms of stealth viruses. The *SWEEP* NLM can be configured to run as a low-priority background process, imposing minimum load on the network, or as a high-priority process to perform spot checks. ***SWEEP for NetWare*** costs £495 for networks of up to 25 users and £895 for a network with an unlimited number of users. The price includes 12 monthly updates. Tel 0235 559933.

Fifth Generation Systems UK is offering **a complete software security package for just £99**. The package, available until the end of this month, consists of *Fastback Plus* 3.0 (backup utility), *DiskLock PC* 1.0 (access control and encryption) and *Untouchable* 1.1. (virus control). Tel 0494 442224.

IBM UK is holding a **Virus Hands-On Course** (FA58) on 23rd September. Information from the *IBM Education Centre*. Tel 081 864 5373.

Data-Tech is running a three-day **seminar on Detecting, Removing and Preventing Viruses**, 23rd-25th September. Tel 081 780 2412.

Simon & Schuster International has published the **second edition of *Computer Viruses And Anti-Virus Warfare*** (ISBN 13-036377-4) by Dr Jan Hruska which has expanded to 224 pages and includes new chapters covering network protection and the latest developments in virus programming. Tel 0442 881900.

Butterworth Heinemann Ltd has published the ***Computer Security Reference Book*** (ISBN 0-8493-7712) edited by Dr Keith Jackson, Dr Jan Hruska and Donn B Parker. The book covers virtually every aspect of computer security from 'abstraction' to 'zero knowledge proofs'. The book has 949 printed pages and retails for £90.

Beta-testers have let slip that *Microsoft* (an up-and-coming manufacturer of operating system software for personal computers) is planning to ship **DOS 6 bundled with *Central Point Anti-Virus***. Information from Bill Gates, *Microsoft Inc*, USA. Tel 206 882 8080.

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

VIRUS BULLETIN **©1992 Virus Bulletin Ltd, 21 The Quadrant, Abingdon Science Park, Oxon, OX14 3YS, England. Tel (+44) 235 555139. /90/$0.00+2.50** This bulletin is available only to qualified subscribers. No part of this publication may be reproduced, stored in a retrieval system, or transmitted by any form or by any means, electronic, magnetic, optical or photocopying, without the prior written permission of the publishers.