ISSN 0956-9979 SEPTEMBER 1995

VIRUS BULLETIN

THE INTERNATIONAL PUBLICATION ON COMPUTER VIRUS PREVENTION, RECOGNITION AND REMOVAL

IN THIS ISSUE:

• **All the colours of the Rainbow.** A new virus, Rainbow, has appeared which utilizes circular extended partitions. What does this mean for the user? See the analysis on p.12, and our tutorial on the subject on p.14.

• **Genus and species.** A hoary problem for anti-virus researchers has always been the issue of virus naming. Great efforts are being made to standardise this process, and the first section of a two-part article by Dr David Hull (p.15) clarifies what is involved.

• **Detecting a new way.** *Cheyenne Software* is exploring pastures new; their latest product is *InocuLAN for Windows NT*. How does this product compare with the others in this growing field? Turn to p.18 to find out.

CONTENTS

EDITORIAL

When Techniques Jump Fences

This month's *Virus Bulletin* is perhaps not its usual self. Outwardly it appears the same, but inside, things are different, for it documents not one, but *two* new attack techniques which have appeared in recent weeks and months (see p.8 for an analysis of Winword.Concept, and pp.12-14 for information on the Rainbow virus).

This situation is somewhat analogous to the famous truism of waiting two hours for a bus, and then having three come along at once. New techniques are few and far between, but, like buses, they travel in packs.

"new techniques are few and far between, but, like buses, they travel in packs"

A fairly good working definition of the expression 'new technique' is one which forces anti-virus manufacturers to make some design change to their products. A new polymorphic file infector does not, these days, meet this criterion - the vast majority are very similar, contain nothing new, and (once the producers have updated the virus databases of their products) present no great problem.

Both Winword.Concept and Rainbow meet this criterion, and so will (or should!) provoke some thought from anti-virus producers. Winword.Concept may induce concerns about whether or not to scan *Microsoft Word* files (.DOC and .DOT) - this in itself introduces a world of problems, as the formats of such files are non-obvious. However, Rainbow, which prevents a clean boot, appears to be the more awkward of the two.

The concept of clean booting before attempting to remove viruses is so fundamental to the way the current systems work that a virus which consistently prevents it reliably is bound to cause problems. Rainbow does this on those versions of DOS which are most 'in the wild' (at least in the Western World) - *MS-DOS v5* and above. It is quite within the realms of possibility that a site infected with such a virus would not have clean boot disks of a version earlier than that.

There is a world of difference between an anti-virus product stating that you must have a clean boot disk in order to clean up any infection, and that same product stating that you must have a variety of clean boot disks containing different versions of DOS to suit every occasion. The former is widely accepted, because this is how the system works - there is no real need for a product to deactivate a virus in memory, as a clean boot has always been the simpler course. Although the latter is much more annoying, it is possible that it will be the way people have to move.

In this, as much as in anything else, it is true to say that there is very little which is truly new. The concept of circular partition sectors (*à la* Rainbow) had already been described by the early 1990s, and the idea of a macro virus had been described (albeit in relation to *Lotus 1-2-3*) even before that. However, these techniques have now crossed the barrier dividing the world of research speculation from that of real viruses.

It is interesting to note how long such a crossing has taken - the ideas have been knocked around for so long, and yet have taken this many years to reach the other side of the fence. Well, yes and no: the theories have no doubt been known amongst the virus writers for almost exactly the same length of time as the researchers have known about them.

Whether or not these particular techniques become prevalent in the wild (either by way of the viruses described here, or by other viruses, developed later, which use the same ideas) remains to be seen. However, it does seem highly probable that more viruses using these techniques *will* appear, and this will only serve to highlight the need for anti-virus developers to find ways to make their products deal with them.

One thing is certain - jumping up and down and panicking about the end of the computing world as we know it is not going to help. Neither of these viruses, or their techniques spell doom for the anti-virus industry or modern computing; they simply mean we may have to think about some things slightly differently from now on.

NEWS

C+P+N+A+V = ?

Speculation on the future of *Central Point Anti-Virus* has risen once again, with the imminent release of *Microsoft's Windows 95*. *Central Point Software* was subsumed by the giant conglomerate *Symantec Corporation* last year, and ever since then, industry has been discussing whether or not *CPAV* would be incorporated into the current *Symantec* product, *Norton Anti-Virus* (*NAV*).

Fraser Hutton, a spokesman for *Symantec UK*, has firmly denied the latest round of scuttlebutt, stating that all extant platforms of *CPAV* would, for the foreseeable future, continue to be maintained and supported. He did confirm, however, that the new *Symantec* anti-virus products for *Windows NT* and for *Windows 95* would go under the name of *Norton Anti-Virus*, although they would incorporate some features currently specific to *Central Point Anti-Virus*.

'Our corporate decision has been to continue to maintain and support *Central Point Anti-Virus*,' said Hutton. 'The product is very popular in the market-place, and has strong customer support. There are absolutely no plans to discontinue its production.' ▌

ESaSS and Reflex Announce Alliance

Following the May agreement between *Norman Data Defense Systems* and the Dutch anti-virus software developer *ESaSS BV* (producers of the *ThunderBYTE!* anti-virus utilities), a further collaboration has been announced between the UK company *Reflex Magnetics* (producers of *disknet*, the security package) and *ESaSS*.

With immediate effect, the two companies will integrate their development teams and pool their technology to build their next generation of anti-virus and security products. Each company, through the agreement, gains the right to market the new products throughout the world, with the exception of 'home territory'.

In a press release, John Buckle, Managing Director of *Reflex*, said: 'By combining the technologies of the two companies, we are set to take the market by storm … Through tighter integration of our joint technology, *ESaSS* and *Reflex* are set to become the definitive providers of PC security solutions.'

Dick Gehéniau, vice-president of *ESaSS BV*, commented: 'This strategic alliance will translate our technological excellence into increased market share. This closer working relationship is just the beginning. Expect great things.'

Further information on this alliance is available from *ESaSS BV* (Dick Gehéniau) on Tel +31 889 422282, or from *Reflex Magnetics* (Rae Sutton) on Tel +44 171 372 6666 ▌

Virus Prevalence Table - July 1995

Virus	Incidents	(%) Reports
Form	28	18.9%
Parity Boot	23	15.5%
NYB	13	8.8%
AntiEXE	10	6.8%
Sampo	7	4.7%
JackRipper	7	4.7%
Monkey.B	6	4.1%
AntiCMOS	5	3.4%
One_Half	5	3.4%
Stoned.Angelina	5	3.4%
Junkie	4	2.7%
Viresc	4	2.7%
Leandro	3	2.0%
Bupt	2	1.4%
Stoned.Manitoba	2	1.4%
Stoned.Standard	2	1.4%
* Other	22	14.9%
Total	148	100%

* The Prevalence Table includes one report of each of the following viruses: Amse, Boot.437, She_Has, Cascade-1701, ExeBug.A, Flip, Jerusalem, Jimi, Joshi, K-Hate, LZR, Monkey.A, Natas, Noint, Rex, Stoned.Dinamo, Tequila, Tremor, Trojector, Vacsina, V-Sign, and YMP.

VB '95: Boston on the Horizon

From 20-22 September 1995, the *Fifth Annual Virus Bulletin Conference* will be held at the *Park Plaza Hotel* in Boston, Massachusetts. This will be the first time this highly successful gathering has been held in the United States.

The conference key-note speaker is the highly-acclaimed virus researcher, Dr Harold Highland. Many experts will address a wide range of issues, including the susceptibility of *NetWare*, *Windows NT*, *Windows 95* and *Unix* to virus infection, viruses on the Internet and in a corporate environment, and heuristics.

The two-and-a-half day conference will consist of three streams graded according to technical content, and will also feature an exhibition by security soft- and hardware vendors. The partners' programme will feature a tour of the city, and visits to local sites of historical significance.

The fee for the event is £595 (US$895), and *VB* subscribers qualify for a £50 discount. Information is available from the conference manager, Petra Duffield, on:
Tel +44 1235 555139, fax +44 1235 531889 ▌

IBM PC VIRUSES (UPDATE)

The following is a list of updates and amendments to the *Virus Bulletin Table of Known IBM PC Viruses* as of 21 August 1995. Each entry consists of the virus name, its aliases (if any) and the virus type. This is followed by a short description (if available) and a 24-byte hexadecimal search pattern to detect the presence of the virus with a disk utility or a dedicated scanner which contains a user-updatable pattern library.

Type Codes

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

Amazon Queen.468

CER: An appending, 468-byte virus which installs itself in the Interrupt Vector Table. It contains the plain-text messages: 'Amazon Queen...v1.0', 'WHY?' and 'LoRD Zer0'.

```
Amazon Queen.468    E800 005D 81ED 0300 0E1F 06B4 ACCD 213C 3075 0B2E 3B9E D001
```

Amazon Queen.479

CER: An appending, 479-byte variant with the text: 'Amazon Queen...v1.1', 'WHY?' and 'LoRD Zer0'. The first message may be displayed if an infected program is executed and the virus is active in memory.

```
Amazon Queen.479    0E1F E800 005D 81ED 0500 06B4 ACCD 213C 3075 132E 3B9E DB01
```

Amazon Queen.500

CER: An appending, 500-byte variant with the text: 'Amazon Queen...v2.0', 'WHY?' and 'LoRD Zer0'. The first message may be displayed if an infected program is executed and the virus is active in memory.

```
Amazon Queen.500    81ED 0500 4444 06FF 86F2 01B4 ACCD 213C 3075 132E 3B9E F001
```

Baba.353

CR: An appending, 353-byte variant, named after its 'Are you there?' call: AX=BABAh; Int 21h returns AX=FACCh. It contains the text '=>COMMAND.COM<='.

```
Baba.353            BF00 0181 C646 01B9 0400 FCF3 A45E B8BA BACD 213D CCFA 7503
```

Blue Nine

CR: An appending 925-byte virus with stealth capabilities, which contains the plain-text message: 'Blue Nine Virus by Conzouler 1994'. Of the two known minor variants, B has 'NOP' instructions in its code.

```
Blue Nine.A         50B4 30B9 9A02 CD21 81F9 BC01 7466 3C03 7262 8CC3 4B8E C326
Blue Nine.B         50B4 30B9 9A02 CD21 81F9 BC01 7467 3C03 7263 8CC3 4B8E C326
```

Breeder.4026

PR: An encrypted, 4206-byte companion virus which contains the encrypted text: 'File0000.000 = \RENCODES.BRE'

```
Breeder.4206        8D36 1F01 8BFE 8D16 1F01 8D0E 7D0A 2BCA FCAC D0C8 AAE2 FAE9
```

Diddler.91

CNO: A simple, overwriting, 91-byte virus which infects the first file in the current directory. It contains the text: '*.com Diddler 95 (newbee)'.

```
Diddler.91          0AC0 752D B002 BA9E 00B4 3DCD 2193 B95B 00BA 0001 B440 CD21
```

Diddler.190

CN: A simple, appending, 190-byte direct infector with the text: 'Diddler[Newbie] Evolved *.c?m'.

```
Diddler.190         7242 B43F B903 008D 96BE 01CD 213E 80BE BE01 E974 2F3E 8B86
```

Elaine.1127

CER: An appending, 1127-byte virus which contains the text: 'Elaine 1.0 28 May 1994'. As a payload, the virus hooks Int 13h (functions 03h, 0Bh). When active in memory, it may corrupt data in the write buffer (random changes to the first byte in the buffer).

```
Elaine.1127         B813 35CD 2189 9C1B 008C 841D 00B8 FE4B CD21 3D11 1174 4DB8
```

Fistik

CER: An appending, 1280-byte (COM files) or 1536-byte (EXE files) virus containing the plain-text message 'Dnyalar Tatl', displayed when the virus is active in memory and has infected five files.

```
Fistik              CF3D 004B 7405 2EFF 2E32 012E 803E 3101 0572 03E9 0C02 2E8C
```

Forget.1203

CER: An appending, 1203-byte virus which marks all infected files by putting the byte CCh at the end of programs. In January 1995 it displays the (normally encrypted) message: 'Forget it, I'm lazy today!'.

```
Forget.1203         FCF3 A45E 1F06 B84D 0050 CBB8 43FD BB12 00CD 213D 1256 741A
```

Human Greed.666

ENO: An encrypted, overwriting, 666-byte virus which infects files on drive C. The long message included in the virus body begins: 'That is not dead...' and ends: '...*** HUMAN GREED *** The answer of all evil on earth! Do You Belive? Farwell!'.

```
Human Greed.666     BE2F 018B 1616 01B9 3301 2E31 1483 C602 E803 00E2 F5C3 C386
```

Istanbul.1349

CER: An appending, 1349-byte virus containing the text: 'Anti-Virus??Written in the city of Istanbul (c) 1993' and 'Installed'.

```
Istanbul.1349       3D24 4675 04B8 3434 CF3D 004B 7402 EB6E 5156 5706 5053 521E
```

John

CN: Appending, 1962-byte, direct, fast infector. It displays at random two screens of information on John Buchanan (better known as Aristotle). Infected files start with the plain-text message: 'Ari is a NARC'.

```
John            81BE 8A08 4D5A 7437 81BE 8D08 4172 742F B802 4233 C933 D2CD
```

Maresme

ER: An appending, 1062-byte, encrypted virus containing the text: 'Virus Maresme Show by XUTE !!!'.

```
Maresme         0003 F38B FE8B 9711 03B9 E603 AC32 C22A C2CD 01AA CD1C E2F4
```

Milikk

CR: An appending, 1020-byte virus with stealth capabilities, which corrupts the MBS. The virus remembers how often an infected file was executed and keeps the counter inside the MBS of the first hard disk. After 150 infections, it overwrites the boot procedure with its own code. When the system is next started, the text 'M I L I K K' appears in the centre of the screen. After a keystroke, the operating system is loaded as usual.

```
Milikk          E800 005E B8F4 FF81 EE46 04CD 213D 0B00 7503 F972 180E 1F0E
```

Ohlala.1960

CEN: An encrypted, appending, 1960-byte, direct infector which infects six files at a time (three COM, three EXE). It contains the encrypted text: 'Ohhhh La La! Mommmmy, they are teasing me again Shut up you little sonsuvbitches' and '*.MS *VIR.DAT COMMAND'.

```
Ohlala.1960     BB00 002E 8A04 2E30 8129 002E 8A81 2900 89FE 29C6 434E E2EB
```

OS.840

CR: An appending, 840-byte virus which marks all infected files with the string 'OS' placed at the end of programs. It contains only one ASCII string: 'c:\command.com'.

```
OS.840          80FC FF75 03B4 FECF 3D21 2575 01CF 3D00 4B74 03E9 AA01 5053
```

RiP

CR: An appending, 3214-byte virus with the plain-text messages: '>-[RiP]-<' and 'RADICAL_iNVADiNG_PARASiTE (RiP)-ViRUS, iN 94/95 BY AeMISc, SAYZ Hi 2 U!'. When active in memory, the virus infects an executed COM file and one file in the current directory.

```
RiP             B97F 00BE 8000 F3A4 C3B8 8552 CD2F 3D07 0375 03E9 F900 BF39
```

SillyC.140

CN: A simple, appending, 140-byte, fast direct infector. Unlikely to become common in the wild, since it spreads only under DOS 2.11 and when the Country Specifier is set to 2Eh (Sweden).

```
SillyC.140      81ED 0701 8DB6 8C01 BF00 0157 A5A5 B438 CD21 3C2E 7512 B41A
```

SillyC.190

CN: A simple, appending, 190-byte virus which infects one file at a time. It contains the string: '*.COM'.

```
SillyC.190      A300 018A 45FC A202 01B4 1A81 C7B2 008B D7CD 21B4 4E33 C981
```

SillyRC.212

CR: A simple, appending, 212-byte virus which marks all infected files by setting the last byte to 0EAh.

```
SillyRC.212     A5A4 C33D 7742 7501 CF3D 004B 756C 5053 5152 1EB8 823D CD21
```

SillyRC.476

CR: Appending, 476-byte virus, similar to SillyRC.212. It contains the plain-text messages: 'Subconsious virus - Conzouler /IR 1995' and 'Mina tankar r det sista som ni tar...'. It also hooks Int 08h and displays for a moment every seven seconds the text: 'LOVE LOVE LOVE LOVE LOVE LOVE LOVE LOVE'.

```
SillyRC.476     4F56 453D 7742 7501 CF3D 004B 756C 5053 5152 1EB8 823D CD21
```

Sofia.432

CR: An appending, 432-byte virus which installs itself in the Interrupt Vector Table. It contains the plain-text messages: 'This Virus is named after a very nice, clever and cute girl, Sofia', 'Sweden', and 'LoRD Zer0'. The virus creates one hidden, 7-byte long file called 'SOFIA'.

```
Sofia.432       9C80 FC4B 743B 3DBE BE74 1D3D 0378 7512 80FF 1975 0D81 FF4C
```

Sofia.528

CR: An appending, 528-byte variant of the Sofia.432. It resides in the same area, contains the same messages and creates an identical, hidden file. It intercepts two more functions (11h and 12h) of Int 21h.

```
Sofia.528       9C80 FC11 742C 80FC 1274 2780 FC4B 7473 3DBE BE74 553D 0378
```

Taurus.562

CR: An appending, 562-byte virus containing the encrypted text: 'Happy New Year !' The message is displayed in January, every day between 2:30pm (14:30) and 3:00pm (15:00). The virus reinfects already-infected programs, files growing by 562 bytes with each new infection.

```
Taurus.562      B821 25BA C900 1E06 1FCD 211F BF14 033E 8B03 4747 3E8B 1B47
```

TeaForTwo

CR: An appending, 1024-byte virus containing the plain-text message 'T42 Tea for two!' at the end of infected programs. It was written as a multi-partite virus infecting DOS boot sectors on floppies and files. The copy investigated contains a minor bug, so the virus hooks Int 13h, overwriting some sectors but making diskettes unbootable. The bug is easy to repair, so we will probably see a fix in the near future.

```
TeaForTwo       B8FF 25D1 E040 CD21 B425 D0E4 BBFF FFCD 2181 EB80 00B4 25D0
```

VCL.279

CNP: A 279-byte companion virus containing the text: '[VCL_MUT] The Pleasure 2 VirusEver have the pleasure?By eMpIrE-X'.

```
VCL.279         B903 0051 E808 0059 E2F9 58B4 4CCD 21BA 2C01 E807 00C3 2A2E
```

VCL.316

CNP: A 316-byte companion virus containing the text: '[VCL_MUT] The Pleasure 6 VirusEver have the pleasure?By eMpIrE-X'.

```
VCL.316         B903 0051 E808 0059 E2F9 58B4 4CCD 2155 8BEC 83EC 40B4 4732
```

Virogen.1535

CER: Polymorphic, appending, minor 1535-byte variant containing the encrypted text: '(c) 1993 Virogen ASeXual Virus v1.00'. It can be detected in memory with the pattern for variant 1520 (see *VB* July 1995).

INSIGHT

Igor Grebert: Carpe Diem

Igor Grebert belongs to a family whose interest in computers reaches back through two generations. He was born on the French Riviera, and grew up in Paris, though he travelled extensively in Europe and the USA. 'Most of my summers,' he said, 'were spent on the beaches around Cannes; sailing, windsurfing, or fishing for sea urchins.'

Family involvement with computers stretches back to the 1970s: 'My uncle and my father designed their own computer called ALVAN in the early 70s. My uncle, Alain Grebert, headed a team of engineers in Philadelphia: they designed a mini-computer around a new language they had developed. It was the first computer I ever programmed - I was eight.'

This exposure led him to the *TRS80* and the *Apple*: games held no interest for Grebert; he was driven to make machines do what he wanted. Later, Grebert studied at one of France's famous engineering schools, *L'Ecole Centrale de Paris*, where he majored in Bio-technology. His special interest was brain simulation: 'In my opinion, there was something missing in the AI field then, and I wanted to understand better what it was.'

Living in America

Grebert fulfilled his military obligations doing research into pattern recognition through neural networks at *Stanford University* in the US: 'I was working with *Boeing*; playing with ideas on making planes land with an improved version of automatic pilots using neural network techniques.'

A few years prior to this, he had met John McAfee, who was at the time working on a PC voice recognition board - Grebert was handling the application programming of the boards in France. This led eventually to a job offer, addressing user interface issues on the *McAfee* anti-virus product.

'That was fun,' reminisced Grebert, 'but after a few weeks there, he challenged me with the Number_of_the_Beast virus, asking me to write a remover for it. That was the beginning of my involvement with PC viruses.'

Then came 512: 'We call it the Stealth,' he said. 'It's kind of interesting to play with a stealth virus at first - I was pretty foolish that time; I was standing there and telling him, "No, John, it doesn't infect, there is nothing, look at it!". That experience made me learn pretty quickly, and I've been learning constantly ever since.'

He still remembers his first encounter with a customer virus problem, a Jerusalem variant which played *Frère Jacques*: 'It triggered a reaction; it was a challenge. 512 was programming; stuff I played with - suddenly, it was affecting customers, people, companies. It was only then I understood that what we were doing was helping - I mean, that company had nothing to do with viruses; it damaged all their backups; made them lose time. They didn't deserve all that.'

The World of Viruses

Grebert has not seen anything really new for over a year now: 'Every new virus we see today belongs to a category which already exists,' he explained. 'This is a contrast to previous years, which makes me think that virus authors are running out of ideas. I believe there will be little change for the next year or so. Then, probably, we will see a few new techniques, but I do not foresee anything radically different.'

Grebert believes that no single anti-virus technique is sufficient to ensure a virus-free environment. Heuristics alone, he believes, will not allow for detection of existing viruses: 'This is why we offer multiple products, and use multiple technologies in our scanners. I believe that we have already integrated the best part of heuristics in our tools and in our scanner, and are now fine-tuning them constantly.'

Heuristics, in his view, have merit, but one must be cautious as to how they are implemented - the inherent risk is false alarm. The future, he feels, is in the harmonious integration of techniques which allow reliable and generic detection of viruses. He sees the best answer to polymorphic viruses as improving virus-specific detection to enable their detection and identification: 'There are simple ways,' he stated, 'to handle these, which are time-effective, and reliable.'

Ethically Speaking

Grebert has definite opinions on virus-writing: 'There is a dilemma between preserving the right of expression and protection against crimes,' he said. 'One should be allowed to play with such ideas as self-replicating code, as long as the environment is strictly controlled, but no-one should be able to force me to run a program I do not want to run on my own machines. Between the two is a fine line which the legal system has yet to define satisfactorily.'

The very thought of virus-writing is alien to Grebert - his only contact with virus authors is through their creations. He has never created a self-replicating program, feeling his time is better spent doing other things: 'The idea of adding the ability to spread has never struck me as interesting,' he said. 'If I have a message, I can use other means to convey it.'

He professes himself disgusted by the amount of time, money, and effort the world has lost over viruses, and does his utmost to counter this, anticipating what the next threat might be, and preparing programs to handle them as soon as possible. 'To do this I do not need to write any such code,' he explained. 'I simply explore the OS internals.'

Igor Grebert is a rarity for a virus researcher; having just as many interests outside work as in!

Professional Growth

Since 1989, Igor Grebert has worked at *McAfee Associates*, an organisation which has recently acquired many smaller companies. Grebert is quick to stress that acquisition played a much smaller role in the deals than development: '*McAfee* is growing out of the anti-virus business towards network management,' he explained. 'Most of our installed base was in companies with networks; people trying to implement anti-virus policies had other problems to address - software distribution, application metering, remote desktop control.

'There are many anti-virus companies around,' he continued. 'It is no longer easy to start a company with no international presence, but new developers can still prove themselves. They have to do this in concert with existing companies, though, as the industry has grown so much. Writing an engine is still fairly easy, and ideas can easily be implemented and tested, but the package is more than the engine.

'You have to support multiple platforms, build interfaces, think network, and client/server. The same thing applies to people who want to write a new OS... What was possible ten years ago is not today - but new opportunities are available today that did not exist then.'

Always, at the core of Grebert's work, are viruses: 'I wanted to work on detection of the "weird" viruses, and... I've always been obsessed with the idea of finding something that would allow me not to work any more. If you're a good programmer, you don't want to waste time, to do things two or three times. One thing you try to do is to automate as much as you can, and to make your scanners as good as possible, so you just push a button to detect the latest virus.

'The technology we had did not allow us to do that - we all have to change some time. What keeps me going at *McAfee* is the opportunity to change technology, and to redesign the scanner from the ground up. As John worked on making the company grow, he allowed me to take on technical leadership; managing the anti-virus researchers and programmers.'

In the Office

Grebert is currently Manager of Research and Development at *McAfee*: 'One of many!' he laughed. 'The anti-virus stuff is what I've been focusing on, but we have network management, we have utilities for *Windows*, we have a replacement for the shell program, and so on...'

Grebert's brief is to find better ways to handle viruses, or to automate the way in which they are processed: 'We retired the older version of our product, and are moving towards a new, more compatible version that goes across platforms, that requires less work from the programmers,' he explained. 'We don't have to rewrite the *Windows* or the *OS/2* parts - it's all integrated, and makes for a very easy-to-use development platform. That was the challenge for our team.'

There are still challenges, however - integrating his knowledge of viruses to a point where the process of detection and removal is almost automatic: 'It's what we have to do! The scanner is the ultimate holder of the technology you've put together. We want the amount of work that has to happen to look at an ordinary virus to be no more than about an hour.

'This is inside a development scheme: you receive the file, someone looks at it, another answers the customer: there's a whole process. The amount of work (granted the virus infects nicely) is a few hours, including removal. When it starts to use techniques which are a little hairier, you need a little more time - but I believe this too can be automated.'

Inside Outside

Though Grebert admits that he was once a 'pizza-and-coke' programmer who routinely worked 80 hours a week, he does now take time out: 'I enjoy going away. I've just come back from Lake Tahoe - it's only a few hours from the Bay, so it's somewhere to go for the weekend. When I travel on business, I often end up spending the weekend in various cities. I like to windsurf - there are places here where I can do that.'

There are still times when he has to work 'from sun-up to sun-down', but Grebert insists that this is not a healthy approach in the long term: 'You cannot do this for four or five years running and still keep your peace of mind.'

Of course, as a Frenchman, one of Grebert's great pleasures in life is food, from sushi to hamburgers ('But you cannot eat hamburgers every day!' he insisted). He enjoys cooking for himself and his friends, and going out to good restaurants: 'There *are* good restaurants here,' he avowed. 'You just have to find them, and be ready to pay the money.'

He does miss France, however; the good food and the cheese (this latter he finds difficult to obtain in the USA) - one day, he says, he will return, but not before his work at *McAfee* is finished. In the meantime, between skiing at Lake Tahoe, and having a house which, in his words, often resembles an international hotel with friends from Australia, Japan, and Europe always around, Igor Grebert remains a man who seizes every day.

VIRUS ANALYSIS 1

What a (Winword.)Concept

Sarah Gordon
Command Software Systems Inc

Winword.Concept is a remarkably friendly virus, which happily infects across platforms. Yes, that's right, *Macintosh*, *MS-DOS*, *Windows NT* - if it runs *MS Word*, it can be infected. Thus, people using mail interfaces which make use of the *Word* application can get a virus by reading electronic mail. The statement 'You cannot get a virus by reading your mail' is no longer true. You can.

Perhaps calling the techniques used by this virus a 'new concept' is not totally accurate. We knew this type of vulnerability in a macro language would be exploited sooner or later. Perhaps we can consider ourselves fortunate that the virus has no destructive payload: its only obvious problem is an inability, in some cases, to save work - it could be worse.

Apparently non-malicious in intent, Winword.Concept nevertheless introduces us to a new threat. In the past, we have seen fast infectors, polymorphics, stealth. This virus merely uses incredibly simple techniques to replicate and hide from the user, once a file is infected.

The appearance of this virus presents anti-virus product developers with a challenge in implementing detection, as, rather than spreading by infecting more traditional types of 'executable' code, it adds itself as a small macro to *Word* templates. This allows the virus to infect and spread utilising files with any extension; as long as they are in *Word* format.

An Operating System by Any Other Name

As applications become increasingly complicated, they have begun to resemble mini-operating systems, supporting their own little file system and command set. *MS Word* has its own programming language, *WordBasic*, which, as the name implies, is reminiscent of 'real' BASIC. Although programming with *WordBasic* is not described in the *Word* manual, further information can be obtained by using the on-line help facilities, or by ordering the *MS Word Developer's Kit*.

Thus, every document has the potential to carry code which represents 'executable' instructions in the *Word* environment. However, this still doesn't explain how these instructions come to be run. After all, even if a document contains a set of macros, they have to be explicitly run, right?

Wrong.

AutoOpen = AutoInfect

In its default configuration, whenever *Word* opens a document, it searches for the presence of a macro named AutoOpen and executes its contents. This is carried out without asking or alerting the user, and so is usually a completely transparent process. The user is aware only that he has successfully opened another document; another triumph of the computer age!

In general, the AutoOpen macro will set up the working environment required by the document or the user. However, *Word* has no concept of privilege and allows the macro to make permanent changes to the way it functions. This is a powerful and useful feature, and one which is open to a great deal of misuse.

In the case of Winword.Concept, the AutoOpen macro first checks to see if the virus is already active on this computer, by searching the environment for the presence of a macro named 'PayLoad'. If this is present, execution aborts.

A second check is made for the presence of a macro named 'FileSaveAs'; if found, the virus sets an internal flag, and again aborts infection. The internal flag used by the virus to signify this is called 'TooMuchTrouble', possibly indicating that if the user already has a macro named 'FileSaveAs', it is simply too much trouble to continue and infect the system.

If these tests are passed, the virus adds four new macros to the user's 'global document template'. This is stored in a file named NORMAL.DOT, and is a general purpose template for any document.

To quote from the *Word* manual: 'Unless you select another template when you create a new document, *Word* will base the document on the Normal template.' The four new macros are AAAZAO, AAAZFS, PayLoad and FileSaveAs (the contents of the FileSaveAs macro are simply copied from the virus' macro AAAZFS).

The virus displays a dialog box upon infection, containing what appears to be an infection counter, but which displays the number '1' no matter how many infections you generate. On examination of the macro code, it is observed that this is due to sloppy programming on the virus author's part.

Once this message box is clicked on, the virus is resident, and execution of its 'bootstrap' macro finishes. Once resident, the virus code is activated whenever the user attempts to save a file using 'File/Save As', as this function has been 'enhanced' by the addition of a FileSaveAs macro. Whenever the user selects this option, the virus creates an AutoOpen macro in the new document, and copies the contents of the macro AAAZAO into it. The macros AAAZFS, AAAZAO and PayLoad are also created and copied into the new document.

Thus, the virus code is added to all those documents which are stored using File/Save As, and it is ready and waiting to spread when that document is sent to another unsuspecting

user. There are two things worth noting: the macro called 'PayLoad' is never executed, and it contains only the following text:

```
Sub MAIN
    REM That's enough to prove my point
End Sub
```

The name of this macro is not an empty threat: examination of the virus code and the *WordBasic* language shows that it would require a trivial alteration to make the PayLoad macro active and to give it a wide variety of different functions.

Detection and Removal

Checking whether a copy of *Word* already contains the virus is trivial. Start the program, and select the Macro option under the Tools menu, choosing Macros Available in 'All Active Templates' option.

This displays a list of macros currently installed on the computer; if AAAZAO, AAAZFS, FileSaveAs, and PayLoad are present, the machine is infected. Highlight each of the virus' macros in turn and select the Delete option. This removes the virus, but does not solve the problem of the infected files on the system.

There are other ways to detect this virus in files. One is to add user-defined virus strings to anti-virus programs which have this feature. The user can add '3A 41 41 41 5A 41 4F' and/or '3A 41 41 41 5A 46 53', scanning all files. These scan strings are the hex representation of the ASCII strings ':AAAZAO' and ':AAAZFS', and will be found in any document containing that text.

Since .DOT and .DOC files are not typically scanned, it is important to remember to add them to the list of file types to be scanned. If you suspect you have this virus, you may want to scan *all* files, as your users may have changed the filename extensions after saving the files.

Alternatively, you can search every document on your system for the strings (and the rest of the virus) using a disk editor. This could prove a lengthy process and is not recommended.

If you find these strings in a *Word* document, further checks must be made. Unfortunately, these are difficult, as the virus is composed entirely of plain text, making it difficult for someone without knowledge of *Word* to decide whether even a *Word* document which contains these text strings is the virus itself, or a message warning of the virus' presence.

One definitive way to determine whether the document is infected is to open it using *Word*, though this is counterproductive. My suggestion is that if you find the macros listed above active within *Word*, call your anti-virus software vendor, who should be able to talk you through a fix.

You can restore infected documents to their pre-infected state manually. To do this, with your infected document loaded, do the following:

- use Edit/Select All to mark the whole document; then Edit/Copy to copy the document to the clipboard
- create a new, untitled document using File/New
- using Edit/Paste, place the contents of the clipboard into the new document
- close the original document using File/Close
- if you are certain that the new document is identical to the old, except for the missing virus macros, use File/Save (*not* File/Save As) to save the new document over the old
- if you are not certain the new document is identical to the old, use File/Save to save the new document with a new name, keeping the infected document isolated in a safe place until you are sure you no longer need it

Manual removal of the virus via other methods is best performed by someone experienced in *Word* document structure.

Automated detection and removal of the virus is offered by several vendors, including *Command Software Systems*; its fix, Wvfix.zip is available free of charge from the *Command/F-Prot* library section of the *NCSA* Anti-Virus vendor forum on *CompuServe,* or via anonymous FTP from ftp.commandcom.com (questions/comments may be mailed to winword@commandcom.com, and will probably end up in my mailbox).

The Problem; the Solution

The techniques used by this virus are so simple that any idiot could use them to construct similar viruses. If history is an indicator, we can expect to see more of this type of virus.

While a short-term fix is available, the ease of creation and modification means that we must find a long-term solution to this general threat. As far as I can see, the most likely way will be to alert the user to any changes made to his global settings. While this will not prevent such a virus from spreading, it will provide users with some warning before their application is reconfigured.

Security is no longer the realm of the OS developer; application programmers should keep a careful eye on the possible misuse of the extra functionality they are providing.

Winword.Concept

Aliases:	Word prank macro.
Infection:	*MS Word* documents.
Self-recognition in *MS Word* documents:	Searches for a macro named 'PayLoad'.
Trigger:	None.
Removal:	See text.

VIRUS ANALYSIS 2

Byway: The Return of Dir_II

Dmitry O Gryaznov
S&S International plc

Those people who have been interested in computer viruses since the early 1990s may remember the 'pancomputeria' caused by the Dir_II virus in the autumn of 1991 - this was a virus which swept around the world like wildfire.

History of the Technique

An infection technique which was completely new at that time was introduced with the advent of the Dir_II virus, and made it the fastest infector ever. In fact, Dir_II brought with it a completely new category of computer viruses: file system infectors.

The virus installs itself as the main DOS disk driver, and intercepts all disk accesses to floppy or hard disks. Then, on any disk access, Dir_II scans the data being read or written for possible disk directories.

If the data reveals a directory, the virus modifies all directory entries referring to executable (COM/EXE) files to point to one and the same cluster chain where the virus has stored its body. The original start cluster number of an infected file is stored, encrypted, in the unused parts of the DOS directory entry.

When the virus is memory resident, everything appears normal, since the virus intercepts any directory accesses, modifying the images of directory entries in memory to their condition before infection.

When there is no virus in memory, however, DOS 'sees' the actual state of directory entries as they are stored on the disk. In this case, since all the executable files are cross-linked to the same cluster, running any executable file results in the virus being loaded to memory and executed.

Strictly speaking, Dir_II does not infect files - the file data, as well as its cluster chain, remains unchanged. The virus 'infects' directory entries instead, cross-linking them to the single cluster chain containing the virus body. So, if you boot a computer from a clean DOS diskette and run CHKDSK on an infected disk, CHKDSK will report dozens of files cross-linked to the same cluster, as well as dozens of lost cluster chains.

With the virus in memory, however, everything looks fine. Since Dir_II intercepts disk accesses at a DOS driver level, presenting itself as the main DOS built-in disk driver, just about any disk access will enable the virus to replicate. Simply typing DIR is sufficient to enable the virus to infect all the executables in the directory from which you requested a listing.

If you accidentally type WIM instead of WIN, DOS will look for an executable file named WIM.COM (or WIM.EXE, or WIM.BAT) not only in the current working directory, but in all the directories listed in the PATH environment variable as well. The result is that all the executable files in each of these directories will be infected by the virus.

This infection technique enabled Dir_II to propagate with unparalleled speed. First released in Bulgaria, it took Dir_II only several weeks to become the most widespread virus in the world in the autumn and winter of 1991.

Fortunately, it did not last long. Dir_II is now believed to have been extinct in the wild for some time, mainly because it appeared to be incompatible with DOS versions 5.0 and above. The memories of this virus survived, making Dir_II a sort of anti-virus 'scary legend'. Yet recently we have faced a 'reincarnation' of Dir_II, in the form of a virus called Byway or TheHnd.

"unlike Dir_II, however, Byway operates pretty well even under the latest versions of DOS"

Dir_II Reincarnate

Byway uses the same extremely fast and effective infection technique which was introduced in Dir_II. Unlike Dir_II however, Byway operates pretty well even under the latest versions of DOS, a fact which might well make it the Dir_II nightmare of today.

To make things even worse, Byway is a polymorphic virus, changing its appearance from one infected disk to another. Its code is written in an extremely obfuscatory manner, with many self-modifying instructions and unusual addressing modes. All this helps make its disassembly and analysis anything but a piece of cake.

Stealth Capabilities - Not Quite There

Still, there is a flaw in this otherwise next-to-perfect virus: its stealth capabilities. To protect the cluster chain where the virus body is kept, Byway creates a 2048-byte-long file called CHKLISTx.MSx in the root directory of an infected disk. The character 'x' in the file name represents the non-printable ASCII code 255 (0FFh), which is displayed onscreen as a space.

The file has System, Hidden and ReadOnly attributes set, so it cannot be viewed by a simple DIR command. You can, however, use the DIR command '/ASH' to see the file. The

switch '/A' forces DIR to show files with particular attribute bits set; the switch '/SH' specifies System and Hidden respectively. So, if you see a file called 'CHKLISTx.MSx' with these attributes, your computer is likely to be infected with Byway!

Text Strings and Trigger

In other ways, the virus is functionally very similar to Dir_II, although, judging by its disassembly, it was an independent 'project'.

The text strings: '<by:Wai-Chan,Aug94,UCV>' and 'The-HndV' are found inside the encrypted virus body. The former, slightly altered, gives the virus its name of Byway, though variations on the first (TheHnd) are also used.

Starting in 1996, providing the day of the month is equal to the doubled month number plus two (i.e. 4 January, 6 February, ..., 26 December), the virus may trigger while infecting a computer.

When triggered, Byway displays a scrolling text phrase, 'TRABAJEMOS TODOS POR VENEZUELA!!!', accompanied by a tune which might well be Venezuela's national anthem. The phrase itself is Spanish for 'Let us all work for Venezuela!!!' or something close to it - I do not speak Spanish myself, alas.

We at *S&S International* are currently receiving an increasing number of technical support calls regarding Byway. Unfortunately, they prove the prediction that the virus is quickly becoming very widespread - exactly like its forerunner, Dir_II.

Detection and Repair

Fortunately, several anti-virus products are already capable of detecting this virus. As for repair, the method used to remove Dir_II also works well with Byway. This is, basically: 'Let the virus disinfect itself', a strategy which works not only for file system infectors, but for full-stealth viruses as well.

The removal method is based on the fact that a stealth virus effectively 'removes' itself from a file being read. The word 'removes' is in quotes because a virus does not necessarily remove itself physically from the file, but rather returns the image of the file in memory to the condition in which it was before infection.

So, if an infected file is copied to a place which a stealth virus cannot infect while the virus is active in memory, the copy will be virus-free. In the case of both Dir_II and Byway, it is enough to PKZIP (or ARJ, LHA, etc) all the files on an infected disk while the virus is active in memory, then boot from a clean system diskette, reformat the disk, and restore the files from the archive. Due to Byway's stealth technology, file copies which are placed within the archive will be disinfected.

Also, since the virus infects at the DOS driver level, it is not able to infect any files on a *Novell* (or, for that matter, any other) network file server. So, it is possible simply to copy all the files from an infected workstation (whilst having the virus active in memory, mind you!) to a server, reboot the workstation from a clean DOS floppy disk, reformat the local hard disk, then restore all the files from the server to the workstation.

The third possibility would be to back up the contents of an infected disk to a tape on a 'dirty' machine and to restore them to the reformatted disk in a virus-free environment.

There are at present two slightly different variants of Byway known. They contain somewhat different encrypted text messages, but are functionally virtually identical. Therefore, both detection and disinfection methods described above will work for either of the two variants.

Byway

Aliases: Dirll.TheHnd, DIR2.BYWAY, DIR.TheHnd.

Type: Polymorphic, memory-resident, encrypted file infector with stealth capabilities.

Infection: All executable files.

Recognition:

The DOS command 'DIR /ASH' shows a 2048-byte-long file called CHKLISTx.MSx, with System, ReadOnly, and Hidden attributes set, in the root directory of the infected disk.

Self-recognition in Files:

Compares the starting cluster number to that of the virus.

Hex Pattern in Files:

```
8BF0 * 8BFE * FD * 4974 *
AD * 35 * AB * EB
```

(within 28h bytes of beginning of file)

Hex Pattern in Memory:

```
501E 5657 B0F0 BE58 040E 1FFC
06C4 7C1B A4A5 A58B 7C1B A407
```

Intercepts: No interrupts intercepted.

Trigger: Running text message displayed: 'TRABAJEMOS TODOS POR VENEZUELA!!!', accompanied by tune.

Removal: PKZIP (or similar) all files on infected hard disk, boot from clean system floppy, restore hard disk, and restore files from archive.

VIRUS ANALYSIS 3

Rainbow: To Envy or to Hate

Jakub Kaminski

Only a small number of the thousands of viruses written merit analysis. Most researchers do not have the time to go through even those which are 'worth' examining closely. Often, when a virus is detected and cleaned, it is shifted to the 'to-do-in-near-undefined-future' pile. Those which do encourage closer examination are likely to be new, unknown specimens spreading quickly in the real world.

Not long ago, I was asked to check a PC which could no longer run *Windows*, and had problems booting from a floppy. I expected to find corrupted files or sectors, along with disabled boot from floppy, or perhaps something 'Monkey-like' fiddling with the partition table data.

My investigations revealed a 2351-byte, multi-partite virus spreading through partitions and directories, residing in the boot sector and many executable files. Its most interesting characteristics are its stealth techniques, and the method by which it disables clean boot from system floppy without altering the contents of the CMOS. An attempt to start from a system diskette results in a system hang before a command prompt appears - neither drive C nor drive A is accessible.

Infection Symptoms

This virus, Rainbow, infects the MBS of hard disks, DOS boot sector of floppies, COM files, and files with EXE-type structure (EXE, DRV, 386, XTP). It is unencrypted, and named after a plain-text message inside its body: 'roy g biv' (an acronym of the colours of the rainbow).

The virus attaches itself to the end of programs. All infected programs have their time stamp modified; the field containing the number of seconds divided by two is set to 31. On infecting a DOS boot sector, Rainbow changes only 25 bytes at offset 3Eh, adding a jump instruction at the sector beginning. The copy of the original boot sector is kept in the diskette's last sector, and the remainder of the virus code written in the preceding five sectors.

When the MBS is infected, only its initial 25 bytes are changed by the virus. The rest of the virus body is written into five sectors on track 0 (cylinder 0, head 0), starting from sector 2. Rainbow does not keep a complete copy of the MBS: the 25 bytes it replaces are stored in sector 6, offset 142h. It also modifies the MBS in a way which could be described as self-protection or as the payload itself.

The information on the active partition (16 bytes) is copied to sector 6, offset 132h, and the contents of the original Partition Table replaced by this Hex byte sequence:

```
0000 0100 0500 B80B 0100 0000 BC01 0000
```

This is interpreted by the operating system as a non-active, extended DOS partition, starting from head 0, cylinder 0, sector 1; ending on head 0, cylinder 523, sector 56; beginning one sector from the start of the disk, and containing 444 sectors in total. The most important characteristic is that this partition entry points not to another partition but to the MBS itself (head 0, cylinder 0, sector 1). Such a case is often referred to as 'the recursive partition' and can be a big headache to someone using the latest versions of *MS-DOS*.

For users of v5 or v6.x of *MS-DOS*, access to the system containing the recursive partition is no longer possible. Starting from a hard disk or a diskette will put the system in an endless loop in the middle of the boot sequence (the OS loader traces through the extended partition chains and locks itself up, investigating the same sector again and again).

Rainbow incorporates a significant number of system control and stealth procedures. When active in memory, it hooks interrupts 01h (anti-debugging), 12h (hiding 'missing' memory), 13h ('Are you there?' call, stealth/infection of boot sectors), 21h (14 functions used for stealth/infection of files), 24h (stealth), and 2Fh (stealth).

Execution of Infected Files

When an infected file is executed, the virus checks to see if the system is infected, and whether the virus is active in memory. This is done by issuing an 'Are you there?' call (Int 13h, AX=1BADh). The value DEEDh returned in the register AXh means the virus is in control [*'One bad deed', geddit? Ed.*], in which case the original program is restored in memory and its execution follows in the usual way.

If the system is clean, the virus installs itself in memory. It takes the 3K required from the current block of memory (as long as it is the last one in the memory block chain), usually placing its code 3K below the current top of memory. Since the virus relies on the data in the current PSP, it will install itself above the 640K limit if an infected file is loaded high.

Next, the virus hooks Int 01h, and tries to install its own Int 21h handler. Rainbow changes not the Interrupt Vector Table, but the current Int 21h service routine. Installation takes place only if the current Int 21h procedure begins:

```
CMP AH,??
JNBE ??
```

The virus replaces these instructions with a FAR JUMP to its own code, saving the original pointers in the virus code. Then, it hooks Int 2F and installs its Int 13h ('Are you there?' call, response only) handler.

Now, the virus infects the MBS of the first physical hard disk. The Int 13h service routine is modified to include full stealth procedures. Int 12h is then intercepted and a new

procedure installed which hides the 'missing' memory occupied by the virus. Finally, the infected file is restored in memory, and control is passed to the original program.

Booting from an Infected Disk

When the code in the infected boot sector is executed, the virus locates the top of memory, decreases it by 3K, and copies all of its code into the area allocated.

Now, Rainbow installs its Int 13h handler (with all infection and stealth features). This also includes the code to install its Int 21h handler after the rest of the operating system is loaded. The virus relies on checking the address of the Int 24h service routine. If its segment is smaller than 1000h, the virus assumes that DOS is already loaded.

In Memory

When an infected file is executed, Rainbow installs itself in memory, intercepting all subsequent interrupts. Unlike most multi-partite viruses, it does not have to be loaded from an infected boot sector to gain full functionality. Rainbow can spread and infect files and floppy boot sectors even on workstations with no hard disk.

The virus infects diskettes on Read or Write access. When active in memory, it returns the clean, original sector at each attempt to read the DOS boot sector. Files are infected on Execution (Int 21h, function 4Bh), or when opened.

COM-type files are infected only if they are less than 63057 bytes and their extension is COM or com. EXE-type files are infected when file length is as specified in the EXE header. Rainbow's stealth procedures include hiding the length of infected files and the virus signature in the file time stamp.

As self-recognition in files is based on the time stamp, attempts to execute a clean file with a time set to 62 seconds often results in a system crash: the stealth procedure tries to disinfect a clean file, but corrupts it instead. It is the only serious bug (minor, in comparison to the poor coding in the vast majority of viruses) which I found in its code.

Booting Clean

The safe removal of any virus from an infected system is always based on a clean boot from a system diskette, something which, in this case, is not always easy. Those still using *MS-DOS* v4 or lower can use the usual system floppies, but those who upgraded to v5 or higher may find themselves in trouble if Rainbow infects their machines.

To gain access to an infected/corrupted MBS, eradicating the recursive partition problem, either boot from an older version of DOS, or boot from an infected disk, then disable the virus in memory or avoid its stealth routines.

If the former is chosen, a system floppy which has an older version of DOS is required - but how many laptop users have a bootable, 3.5-inch DOS 4 diskette? Diagnostic diskettes which boot to their own operating systems can also help in gaining access to a disk which has a recursive partition problem.

The latter solution requires an anti-virus product which can detect and disable viruses in memory, or can work properly when viruses are active in the system. In the case of the Rainbow virus, this does not appear to be a simple task.

Conclusion

One of the plain-text messages inside the virus body is: '*4U2NV*', which can be read as: 'For you to envy'. Some virus writers may certainly envy the author of Rainbow his ideas and skills, but if this virus becomes common in the wild, the majority of the PC community will only hate him.

Rainbow

Aliases: None.

Type: Multi-partite, stealth, COM/EXE/MBS/DBS infector.

Self-recognition:

MBS: word 83A5 Hex at offset 15h.
DBS: word 83A5 Hex at offset 53h.
Files: seconds field in time stamp = 62.

Hex Pattern in MBS:

```
BB00 7C8E D38B E38E C3B8 0502
B902 00BA 8000 CD13 9AA5 8300
```

Hex Pattern in DBS:

```
BB00 7C8E D38B E38E C3B8 0502
B9?? ??BA 0001 CD13 9AA5 8300
```

Hex Pattern in Files and Memory:

```
E800 005E 83EE 03B8 AD1B CD13
3DED DE75 450E 1F81 C664 0781
```

Intercepts: Int 01h, anti-debugging; Int 12h, hiding missing memory; Int 13h, boot sector infection/stealth; Int 21h (functions 11h, 12h, 3Ch, 3Dh, 3Eh, 3Fh, 40h, 42h, 4Bh, 4Eh, 4Fh, 57h, 5Bh, 6Ch), file infection/stealth; Ints 24h/2Fh, stealth.

Trigger: Recursive partition in infected MBS.

Removal: MBS - boot clean from DOS 4 or lower, replace first 25 bytes with the bytes from sector 6 offset 142h, replace recursive partition data with 16 bytes from sector 6 offset 132h. Alternatively, boot from infected hard disk and disable virus in memory before repairing MBS. Files - although cleaning infected files is relatively easy, to remove virus safely, repair MBS, boot clean and replace infected files with a clean backup copy.

TUTORIAL

Circular Extended Partitions: Round and Round with DOS

Mike Lambert

On pages 12-13 of this month's *VB* is an analysis of Rainbow, a virus *VB* first mentioned in its July 1995 *IBM PC* Viruses (Update). The entry states: 'A system with an infected MBS cannot be booted from a clean system floppy if the machine is running any DOS version of 5.0 or higher'. When the virus was brought to my attention, I thought of the paper I co-wrote with Charlie Moore, 'Circular Extended Partitions: A DOS Vulnerability' (December 1992).

Recognising the Problem

The symptoms of a circular extended partition can be described as follows: when booted, the operating system load hangs and the hard disk access light stays on steadily. The kernel is hung in a loop, reading the same block (or circular chain of blocks) from the hard disk. The solution is to boot a version of DOS without the bug in its kernel.

In the paper mentioned, I published patches for DOS 3.3-5.x (a single-byte patch for each). *IBM* sent me each version of *PCDOS* and asked me to publish a patch for each. *DRDOS* was too complex to patch, so was omitted. *MS-DOS* patches were included in case they were needed in an emergency.

More information on the circular extended partition problem, and a tutorial on DOS disk structures, is included in the paper mentioned above, 'Circular Extended Partitions: A DOS Vulnerability', by Mike Lambert and Charles Moore.

The Rainbow Virus

Rainbow implements the simplest of circular extended partitions. It replaces the entry describing the bootable DOS partition in the Partition Table with a phoney extended partition which points to the MBS. The virus 'stealths' the MBS reads so that, when the virus is resident, DOS sees the correct DOS partition entry and the OS comes up normally. When the virus is not resident, DOS versions which have the circular extended partition bug will hang when booted.

The circular extended partition in Rainbow does not hang *MS-DOS* v3.3 or v4.01 - these can be used to boot today's systems (Rainbow does not work on older CPUs) in the event that an *MS-DOS* v5 or v6.x system does not boot.

To remove the virus, it is necessary to clean-boot a version of DOS which does not have the bug, then restore the MBS from a backup copy. The system should then be rebooted from the floppy (so that DOS will see the DOS partition), and infected files should be replaced.

Circulating a Fix

While circular extended partitions were a problem for all *Microsoft*, *IBM*, and *DRDOS* versions implementing extended partitions until December 1992 (v3.3-v5), the issue should pose no problem to the latest versions - Charlie Moore and I notified all three operating system developers in September/October 1992.

Our paper identified a coding error which results in the problem (this was confirmed by *IBM*). *IBM* and *DRDOS* were happy to hear about the problem, and promised to correct it in the next version.

Microsoft proved to be difficult to contact and did not return calls, faxes, or a message on the *MS-DOS 6.0* beta test hotline. A subsequent article by another author brought the problem more directly to *Microsoft* technical staff via the Public Relations office.

DOS Version 6.x

Curious to explain the note in July's *VB*, I assembled v6 of *MS-DOS* and *PCDOS* products and did some testing. True to their word, *IBM* had corrected the problem in *PCDOS 6.1* (no problem with *PCDOS 6.3* either). Testing the *Microsoft* version 6 series explained the note.

Microsoft v6.0, *v6.2*, *v6.21*, and *v6.22* all still have the same bug in IO.SYS, meaning that *MS-DOS* v3.3 to 6.22 (*PCDOS* v3.3 to 5.02, and *DRDOS* v6.0) will not boot in the presence of a circular extended partition. *IBM* v6.1 and v6.3 do not have the bug. As I have been unable to test with the latest version of *DRDOS*, I do not know if the problem has been corrected as yet.

MS-DOS 6.x Patches

The only responsible thing to do is to publish the patches for the *MS-DOS 6* series in case there should ever be a need to recover an *MS-DOS* system from such a problem. The patch is exactly the same for each version of *MS-DOS 6.x*. Within IO.SYS, the procedure is:

1. Search for bytes 07 72 03 - these are at offset 2918h.
2. Change 03 at offset 291Ah to 06.
3. Write the change back to disk.

I have tested each patch, and all work as intended. The decision to use the patch to bring up a system crippled with circular extended partitions lies with the individual.

If Rainbow ever makes it into the wild, it might be a good idea for *MS-DOS* users to have a disaster recovery floppy without the bug (*IBM v6.1* and *v6.3* do not have it) until *Microsoft* applies fixes to *MS-DOS*.

FEATURE

Computer Viruses: Naming and Classification

David B Hull, PhD
National University, California

The literature of computer viruses is steeped in biological analogy. Even their choice of name, virus, is a direct analogy to biological organisms. The writers of this pernicious code also use this analogy: witness the Dark Avenger's Mutation Engine. Indeed, some parts of the community have gone so far as to suggest the concept of artificial life for these and related creations [*Ludwig, 1993; Stojakovic-Celustka, 1994*].

This paper is an extension of the analogy to the problems of naming and classifying computer viruses. These two issues are problems which are critical to working with living and non-living creations. The need for precise name and classification is rooted in the need to communicate effectively about the item in question. This is true regardless of whether the creation is man-made, such as a Mozart sonata, or natural, such as a lemur.

Naming

Naming involves the development of a set of protocols for creating an acceptable name for any given item in the set under review. The more universally accepted the naming protocol, and the more widely it is used, the more valuable it will become.

Modern zoology has benefited greatly from the adoption of a uniform code: *The International Code of Zoological Nomenclature* [*ICZN, 1964*]. This is a remarkable work, and I recommend it as a model of solutions to issues faced by current virus and anti-virus researchers. It derives from the work of Linneaus, the 'father' of modern biological nomenclature, and in particular is founded on the tenth edition of the *Systema Naturae* published in 1758.

The *ICZN* presents several underlying principals which need to be addressed. First, following Linneaus, it uses a binomial nomenclature; that is, a genus and species name together identify an animal. This can be supplemented as needed with names for Family, Order, etc. However, the Code does not define exactly what a species or genus is.

Second, it establishes a protocol for creating and emending zoological names, which in this case are in Latin or pseudo-Latin and Greek or pseudo-Greek.

Third, it uses the rule of priority (i.e. that the chronologically earliest-recorded name will take precedence) to impose order among conflicting claims about the correct name. The *ICZN* has developed and refined this naming framework. The exact requirements for a valid publication of an *ICZN* name are beyond the scope of this work, but they are certainly worth studying.

Fourth, it ties the name of the species, or genus, to a type specimen. The code does get rather involved here, because this concept is critical to the whole naming process. The important points to note are that the name is tied to a particular specimen, and that this specimen is available to other professionals in the field to examine and compare with other material.

The types must be deposited in a museum or other institution: 'Every institution in which types are deposited should (1) ensure that all are clearly marked so that they will be unmistakably recognized; (2) take all necessary steps for their safe preservation' [*ICZN, 1964*].

Naming protocols are, however, basically independent of a commitment to an underlying organizational structure of the organisms being studied. Indeed, Linneaus had no particular underlying philosophy about the mechanisms and organizational structures underlying what he named [*Hull, 1973*].

Classifying

Classification involves grouping the items in the set under review into categories. In many cases, such as zoology, these categories are nested hierarchically. Classification does involve an underlying philosophy about the mechanisms and organizational structure of the items and groups being classified. This philosophy is also strongly influenced by the purpose for which the classification is to be used.

The division between phenetic, or structural, classification and phylogenetic, or evolutionary, classification has a long and deep history in zoology, for example [*Heywood & Mcneill, 1964*]. The classification of Shakespeare's works for library retrieval as contrasted with literary analysis to determine authorship provides an even starker contrast.

Basically, classification approaches may be divided into three categories: heuristic or morphological groupings aimed at simple assessments of similarity; phylogenetic groupings aimed at tracing evolutionary relationships; and functional classifications grouping by categories of action.

Classifying a killer whale *Orcinus orca*, a gray wolf *Canis lupus*, and a great white shark *Carcharodon carchiaras*, must produce very different groupings with each approach. Gross morphology might group the whale and the shark together as torpedo-shaped sea animals, in contrast to the wolf. Phylogeny clearly would group the whale and the wolf together as mammals against the shark. Functionally, all three are high level carnivores!

Beyond the question of the philosophy underlying the grouping is the issue of actualization. The type of data to be gathered, and the means of developing groupings from the data, is critical to the success and usefulness of the classification schema. Information needed for developing evolutionary relationships is often not available directly and must be inferred from other data. Even the choice of method to create groups can have significant impact on the results; viz the differences in different mathematical clustering techniques on the same similarity matrix data. These choices affect the usefulness of the classification system.

Creating a field guide relies on distinctive features used in the grouping methodology. This holds true whether it is monkeys or missiles being identified.

Current Naming and Classification of PC Viruses

Let us start by examining how viruses are currently named, using the results of the naming committee of the *Computer Anti-virus Research Organizations* [*CARO, 1991*]. *CARO's* classification follows a hierarchical format, based on structural similarity of virus code [from *ftp.informatik.uni-hamburg.de:/pub/virus/texts/tests/vtc/naming.zip - 8/20/94*].

The virus name consists of four parts (to be discussed further in *VB*, October 1995), delimited by periods. The underlying classification scheme is explicitly stated to be based on 'structural similarities of the virus' [*CARO, 1991*].

> *"the transplantation of ... code from one virus to another need not represent an evolutionary relationship"*

An important component of similarity is the use of identical sections of computer code in similar viruses. This can come about either because actual sections of code have been copied from a previous version of a virus, or because similar functionality leads to similar code. The use of structural similarity is not absolutely enforced in the *CARO* scheme.

A second major consideration is the length of active code. 'All short (100 bytes of code or less, messages excluded) overwriting viruses are grouped under a Family_Name, called Trivial. The variants in each family are named by their infective length' [*CARO, 1991*].

Functional criteria (resident versus non-resident) and the type of file infected (COM, EXE, MBS or boot sector) also play a part. In an effort to fit all the viruses in the scheme, classification categories for viruses written in high level languages are also represented by a separate category.

The *CARO* effort is clearly aimed at providing a solid and stable naming system for virus-scanning software. However, the exact methodology used to create *CARO's* classification has never, to my knowledge, been presented publicly.

Naming Issues

A major problem in the current nomenclature of computer viruses revolves around the lack of widely-accepted standards. This leads to many communication problems. Perhaps the most obvious (and also perhaps the most amusing) is *McAfee's* 'Genb' and 'Genp' virus - this is their shorthand notation for a generic boot sector virus and a generic hard disk partition virus.

Virus names should consistently and unequivocally name a specific computer virus. The *CARO* scheme is an excellent discussion piece for developing such nomenclature; however, it should be based on structural similarity only. Other considerations, such as mode of action, or the language used to write the virus, are not central to identification.

Furthermore, as Spafford rightly recognizes, the mode of action and programming language used will mark the structure of the resulting machine code strongly, in any case [*Spafford & Weeber, 1992*].

The second major issue in naming involves specifying exactly what the name represents. Unless the name of a virus is specifically linked to a known piece of code, it is never clear precisely what is being discussed. This leads to the type concept used by the *ICZN*.

In zoology, each scientific name is directly linked to a museum specimen, or other known identification of the organism. This is called the type specimen. It is usually held in a museum, and is specially identified as a type, holotype, lectotype, etc, depending on its exact relationship to the name it represents. These type specimens form the key identifiers for a given name. A group takes its name from that of the type specimen with which it is classified.

The third major issue involves establishment of a valid name. The *ICZN* establishes valid names by 'priority of publication'. In its simplest form, this means that the earliest publication of a valid type description of a previously undescribed organism establishes its name.

This requires demonstration that the new specimen is 'different' from all previously known specimens, creation of a valid name under the *ICZN* rules, designation of the type specimen, and publication in a responsible journal. If, on re-examination, the specimen is found to belong in the same group as an organism with another valid name, the earlier of the two names applies to this group.

Classification Issues

There is also a problem of phenetic (structural) versus phylogenetic (evolutionary) classification. In biological classification, the ultimate goal is to develop an understanding of evolutionary, or phylogenetic, relationships. All classifications still begin with phenetic or structural similarities. These phenetic characteristics are weighted to reflect their relative importance as phylogenetic indicators [*Jardine & Sibson, 1971*].

There is a great deal of confusion in virus classification as to the goal of classification. At one extreme, *CARO* is concentrating on recognition of computer viruses - not an unreasonable approach for an anti-virus organization. In many ways, it parallels the approach used by classical zoological taxonomists, and popular field guides to animals.

At the opposite end of the spectrum are classifications focusing on the evolution of computer virus techniques, and on the individuals writing computer viruses.

Bontchev's discussion of the Bulgarian and Soviet virus 'factories' is a classic in this approach [*Bontchev, 1992*]. Gilad Japhet's anti-virus program *CORAL* appears to be developing towards an evolutionary approach, using techniques which appear to be similar to analytical approaches used in this paper [*Japhet, 1994*].

Computer viruses evolve in complex ways not usually encountered in nature. The transplantation of large segments of computer code from one virus to another need not represent an evolutionary relationship, for example. A newer virus may just represent a debugged or patched earlier version. The virus author may have deliberately incorporated parts of other viruses as a short cut, or because the plagiarized code is useful.

If the virus incorporates code generating 'engines', similar code may appear in viruses with no other similarities. Structural similarities deriving from functional similarities likewise derive from several sources.

There are only certain ways to do certain things with a PC running under DOS, for example. Programmers also, like writers in general, have a particular individual style which leads to coding similarities.

Spafford uses the example of the Internet Worm, where the code used linked lists as the primary data structures. It seems that the first class on data structures and algorithms which Robert T Morris took as an undergraduate used LISP: the lesson stuck all too well [*Spafford & Weeber, 1992*]. This makes using zoological concepts such as 'parallel evolution' particularly tricky in analyzing computer viruses.

A second, tricky problem involves defining the unit of classification. In zoology, the essential unit is the species. A phenetic (structural) definition of a species specifies the smallest statistically coherent unit [*Jardine & Sibson, 1971*]. The phylogenetic (evolutionary) definition of a biological species based on the capability of interbreeding does not appear to have much relevance to computer viruses.

Interestingly, a recent article has presented the idea that the definition of a biological species does not have much application to living viruses, either [*Eigen, 1993*]. This article presents the concept of a 'quasispecies', which Eigen describes as: 'a multitude of distinct but related nucleic acid polymers. Its wild type is the consensus sequence that represents an average for all mutations, weighted to reflect their individual frequency' [*Eigen, 1993 p.45*].

Clearly, well-defined viral quasispecies will group, or cluster, under most classification schemes. Such a definition seems to be a far more useful approach for classifying computer viruses.

The second and final section of this paper will be published in the October edition of *Virus Bulletin*. It will be an exploration of these issues using the Stoned virus; an explanation of the Data Set and the methods used, and a schematic diagram of the *CARO* classification of Stoned.

Bibliography

Bontchev, 1992: Bontchev, V (1992). The Bulgarian and Soviet Virus Factories. Bulgarian Academy of Sciences.

CARO, 1991: CARO (1991). A New Virus-Naming Convention. Computer Antivirus Research Organization.

Doolittle, 1990: Doolittle, R F (Ed) (1990). Molecular Evolution: Computer Analysis of Protein and Nucleic Acid.

Eigen, 1993: Eigen, M (1993). Virus Quasispecies. Scientific American (July 1993, pp.42-49).

Ford, 1994: Ford, R (1994). Viruses for Sale, a Dime a Dozen. Virus Bulletin (June 1994, p.2).

Heywood & Mcneill, 1964: Heywood, V H; Mcneill, J (1964). Phenetic and Phylogenetic Classification (No 6). The Systematics Association, London.

Hull, 1973: Hull, D S (1973). Darwin and His Critics. Cambridge, MA: Harvard University Press.

ICZN, 1964: ICZN (1964). International Code of Zoological Nomenclature (2nd edition). London: International Commission on Zoological Nomenclature.

Japhet 1994: Japhet, G. (1994). CORAL - The File Correlator. Ver 1.02 [program].

Jardine & Simpson (1971): Jardine, N; Sibson, R (1971). Mathematical Taxonomy. London: John Wiley and Sons, Ltd.

Ludwig, 1993: Ludwig, M A (1993). Computer Viruses, Artificial Life and Evolution: American Eagle Publications, Inc.

Ludwig, 1994: Ludwig, M A (1994). The Collection: Outlaws from America's Wild West. Ver 1 [program]. Tucson: American Eagle Publications, Inc.

Ludwig, 1995: Ludwig, M A (1995). Personal communication.

Peer, 1994: Peer, Y V d (1994). TREECON. Ver. 3.0 [*IBM PC* program].

Sequences 183: Sequences (Vol. 183). San Diego: Academic Press, Inc.

Skulason, 1995: Skulason, F (1995). Personal communication.

Spafford & Weeber, 1992: Spafford, E. H; Weeber, S.A (1992). Software Forensics: Can We Track Code to its Authors? (No CSD-THR 92-010). Purdue University.

Stojakovic-Celustka, 1994: Stojakovic-Celustka, S (1994). ALIVE (April).

PRODUCT REVIEW 1

InocuLAN for NT

Jonathan Burchell

Cheyenne Software is known and respected for backup and anti-virus products for *NetWare* servers. This month we look at a new product from the company, *InocuLAN for Windows NT*. It has three components: a server element (three 1.44 MB diskettes, one licence disk), one client for DOS/*Windows* workstations (three 1.44 MB diskettes) and one for DOS workstations (two 1.44 MB diskettes).

Documentation

An Administrator guide and a Client guide were included, both of which are substantial, and extremely professionally and attractively produced. In addition to being operational guides, the manuals cover some basic background information on virus symptoms and network protection, and include an appendix covering the more common viruses.

No on-line virus reference is included, but the company has licensed VBASE, an electronic encyclopædia, from *Norman Data Defense Systems*. It is available free of charge to registered users. A list of detected viruses is available within the software.

Server Element

Installation of the server element requires a 486 or higher computer, 16MB or more of RAM, 5MB disk space and *Windows NT v3.5* or above (workstation or server). The software is installed by running set-up in *Windows NT*. The licence diskette must be inserted on installation. It is not required again, but may be used on a re-install: I suspect it is separate only to ease manufacturing and upgrade issues.

The software has three components: server protection, manager or administration front-end, and alert module. The express set-up option installs all components, whilst custom set-up allows components to be installed individually.

The server component is the element which provides the scanning ability, and is copy-protected via the licence disk. The Administrator or Manager module need not be installed on all servers (they can be administered remotely) and may be installed any number of times, including onto workstations which have no server service installed - this makes for great flexibility in server administration. The Alert component is installed on the nominated message centre.

Networking Concepts

Like many of its *NetWare* counterparts, *InocuLAN* allows several servers to be grouped into logical domains. All servers in a domain must be running *InocuLAN for NT*.

The advantage of grouping servers into domains is two-fold. Scheduled scanning need only be set for the master server, propagating automatically to other servers in the domain, as will scanning service information. Also, the master can be set up as the central message centre, allowing reports and logs to be viewed and administered from a central location.

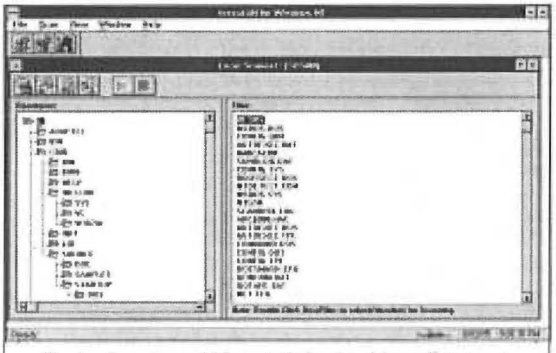
Configuration and administration of all components is accessed via the *InocuLAN for Windows NT* manager icon. The manager consists of three separate modules: the domain manager, the local scanner, and the service manager.

Domain Manager

The domain manager controls and configures domains and scheduled scans. As I had only one *NT* server in my test network, I could not try domain configuration options but, judging from the manual, it is a simple operation to create domains and add servers to, or remove servers from, the domains created.

Domain size may be a single server, or many. A server may be a member of only one domain, and each domain has a nominated master server. As well as domain administration, the domain manger controls scheduled scans, tracking up to 2000 (or 1000 simultaneous) scan jobs. For each, the following information is recorded:

- target drives and directories to scan (a scheduled scan cannot include removable media or mapped drives)
- whether to scan sub-directories of the targets specified
- the CPU usage level (a number from 1-10) at which the background scan is to run
- a list of directories and files to be excluded from the scan (if these do not exist on a particular member of the domain, they will simply be ignored)
- a date and time to start scanning (a repeat interval specified in terms of months, days, hours and minutes)



The local scanner, which controls local and immediate scans, can also access removable media drives and mapped drives.

All files, or executables only, may be scanned. An executable is defined by file extension: the default list is APP, COM, EXE, DLL, DRV, OVL, OVR, PRG, and SYS (BAT and SCR are notable omissions) - extensions may be added or removed. Action to be taken on virus detection includes:

- report only: no action is taken; a message is sent to the Alert module which deals with it as detailed below
- delete file: deletes the file
- cure file: the manual claims that *InocuLAN* can remove, and thus cure, certain infections. It recommends that, even after a cure, you should delete the file and reinstall the original, an attitude we heartily endorse. This raises a question as to whether this option is of use other than if there is no other solution.
- rename file: the default extension for renamed files is AVB (in the event of a file with this extension already existing, *InocuLAN* automatically synthesises an extension of the type AV0, AV! etc). An option allows the default extension choice to be changed.
- move file: moves an infected file to a specified quarantine directory (the default is InocuLAN\Virus)
- purge file: deletes an infected file and guarantees that it cannot be recovered with recover utilities
- rename and move file combines move/rename options

It is also possible to specify scan type: the options are 'fast', 'secure', and 'reviewer'. 'Fast' checks only the beginning and end of a data file, whilst 'secure' checks the entire file and is consequently a little slower. The manual claims that 'reviewer' detects virus-like activity within a file (a heuristic approach perhaps), whilst the on-disk READ.ME file claims that 'reviewer' uses a database of garbage virus strings.

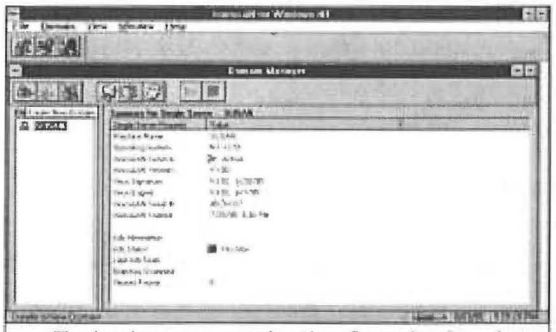
I suspect 'reviewer' contains signatures from test-sets which do not represent true viruses. The *VB* test-set has only genuine, viable infected files. The manual states that using 'reviewer' may cause false positives - I set the scanning to 'reviewer' for the detection tests. Further options in this section allow starting, stopping and rescheduling of jobs.

Local Scanner

The second component of the *InocuLAN* manager is the Local Scanner. This module controls local and immediate scans. Unlike the scheduled scanner, it can access removable media drives and mapped drives.

Options for the scan are broadly similar to those outlined for the scheduled scan, with the exception of job start and repeat information. Additionally, it is possible to request that *InocuLAN* prompts the user before taking any action on an infected file, and that it 'beeps' the workstation speaker when an infected file is discovered.

Selecting what to scan is specified via a graphical tree representation of the drive, which makes it extremely easy to indicate specific directories and files to be included in or excluded from the scan. Unfortunately, I could see no way of saving the choices for another session, or indeed of keeping a list of different types of immediate scan jobs.



The domain manager controls and configures domains and scheduled scans, tracking up to 2000 scan jobs.

Service Manager

The final option in the manager module allows for starting and stopping of the scanning service. This sets whether scanning service starts automatically when a *Windows NT* machine is booted, and sets various parameters affecting how often the service manager should scan job queues, poll apparently dead servers, and hold finished jobs in the queue.

It is also possible, with the event and the scan logs, to set how many messages to retain in the log file (this may be set between 10 and 1,000), after how many days to purge records automatically, and the level of information to be stored. This can be any combination of critical, warning and informational messages. The event and scan logs are accessed directly from within the relevant program sections.

The included *Windows* help files are informative, attractive and easy to use. They offer a dual pane mode, with contents in one screen and the selected entry in another, making it quite simple to 'read' the manual on-line.

Alert

Whenever an *InocuLAN* server or workstation client produces an event (such as detecting a virus), it sends a message to the server nominated as the domain master. There, it is intercepted by the Alert module, processed, and added to the central 'master' database of alerts. A received alert may cause any of the following actions to take place:

- a broadcast message sent to nominated users or groups
- a pager message (numeric or alpha-numeric) sent to a nominated group of recipients. Requires a modem connected to a server machine to access pager service. The message sent consists of a detection code number, a machine ID number and a user-defined custom code.
- SNMP trap messages sent across the network to an SNMP management product such as *NetWare* Management System (NMS) or HP OpenView. Either IPX or TCP/IP may be selected as the transport mechanism.

- Trouble Ticket: this option allows a list of printers to be defined. *InocuLAN* will print a Trouble Ticket automatically when an alert is received.
- Email: this option, which requires *Microsoft Mail*, allows a nominated list of recipients to be notified of alerts via email. It is possible to specify the 'To:, CC: and Subject:' parts of the header as well as to attach a specified list of files to the message (for example, the event log).

The eagle-eyed among you will have spotted that none of the options discussed so far control real-time checking of file read and write. Unfortunately, the interface to the *Windows NT* file system is such a closely guarded secret that no anti-virus vendor has been able to provide real-time file checking for *NT* server products. *Cheyenne* is no exception.

Thus, there can be no real-time protection for server-to-server or workstation-to-server transactions when both systems are using *Windows NT*. As, at the moment, there are no *Windows NT*-specific viruses, this may not be much of an issue. DOS sessions within an *NT* (or *OS/2*) workstation, or on a DOS or DOS/*Windows* platform, may be protected by loading appropriate *InocuLAN* client software.

The critical component is Immune, a TSR which provides real-time checking of files as they are accessed in a DOS session or on a DOS/*Windows* workstation. Immune can send alerts across the network to the Alert master, providing for centralised monitoring of real-time workstation activity.

The Immune/Server communication relies on IPX packets being available as a transport mechanism, which is rather a shame, as many *Windows 95/NT* networks will be NetBEUI or TCP/IP only. However, *Cheyenne* intends to provide support for TCT/IP in the next release of the product.

Results

The main problem with the virus detection provided by the main scanner seems to be the lack of identification of the SMEG and Cruncher polymorphics (plus a slight wobble on some of the MtE variants) and that some basic signature data for the 'Standard' and the 'In the Wild' test-sets is missed. Having said that, however, the detection ratios show the kind of performance which could easily be tuned to 100%.

As is shown in the results table, there are obvious problems with real-time detection. This aspect, represented by the Immune detection figures, is not good enough to guarantee a good level of viral immunity. I suspect that this lower figure comes from the twin pressures of maintaining two code bases and keeping the TSR element for DOS to a reasonable size. *Cheyenne* will shortly be providing VxDs for *Windows* and *Windows 95*, and a similar system for *Windows NT*.

Conclusions

InocuLAN for NT brings the sophistication of big league *NetWare* products to *Windows NT*. It has a user interface which makes the most of the *Windows* Graphical User Interface, and helps ease administration of large networks. The inclusion of features such as domain administration, and sophisticated alert and messaging systems, set it above *SWEEP for NT* in terms of features and may make it more suitable for large sites.

I do have a few gripes, however. The concept of domains, scheduled scans and local scans in the Manager module is a little confused. In a large network, I might also want Alert to function across multiple domains, rather than having to set it up for separate domains.

It also seems surprising that the signature database cannot be automatically propagated to all domain members (or to all members of the visible network). This feature is planned for future release, according to *Cheyenne*.

Having said that, the features and quality of this package are astounding, even more so when combined with the knowledge that this is the first version. Detection ratios, except for some problems with the polymorphics, are good (see results, below), though not as good as those for *SWEEP for NT*.

The good news is that *Cheyenne* feels it will crack the problems of real-time checking on the server. Once this has been achieved, the high detection rates, together with the superb user interface and server administration, mean that this will be a product to consider in any installation for *Windows NT*.

InocuLAN for NT

Detection Results

Main Scanner:

Standard Test-Set[1]	229/230	99.6%
In the Wild Test-Set[2]	120/126	95.2%
Polymorphic Test-Set[3]	3732/4796	77.8%

Immune:

Standard Test-Set[1]	228/230	99.1%
In the Wild Test-Set[2]	118/126	93.7%
Polymorphic Test-Set[3]	1214/4796	25.3%

Technical Details

Product: *InocuLAN for NT*.

Developer: *Cheyenne Software Inc*, 3 Expressway Plaza, Roslyn Heights, NY 11577 USA. Tel +1 516 484 5110, fax +1 516 629 1853, email cheyenne@cheyenne.com.

Price: US$895 (1 server), US$3995 (5 servers), including upgrades (every two months), and licences for all DOS, *Windows*, and *Macintosh* machines connected to the server(s).

Hardware used: Client machine - 33 MHz 486, 200 Mbyte IDE drive, 16 Mbytes RAM. File server - 33 MHz 486, EISA bus, 32-bit caching disk controller, *NetWare 3.11*, 16 Mbytes RAM.

Each test-set contains genuine infections (in both COM and EXE format where appropriate). For details of the Standard test-set, see *VB*, January 1994, p.19 (file infectors only). For details of In the Wild and Polymorphic test-sets, see *VB*, August 1995 p.19.

PRODUCT REVIEW 2

IBM AntiVirus

Dr Keith Jackson

IBM AntiVirus has been reviewed by *VB* several times before: version 1 for DOS in January 1993, the *OS/2* version in August 1993, as part of *PC-DOS* in January 1994, and the *NetWare* version in February 1995.

This review is of version 2.2, which can be used with DOS, *Windows* or *OS/2*. It was provided for review on three 3.5-inch, low density floppy disks. *IBM* claims that its anti-virus software 'is the software that IBM uses to protect its own personal computers' [*I should hope so! Ed.*], and that it is 'designed to detect and remove viruses from your system as simply and reliably as possible'.

Documentation

The documentation took the form of an A4 ring binder, containing 101 pages about its DOS and *Windows* versions. I have no real complaints about the manual - it is readable, well-indexed, explains the basics well; however, it does lack some explanation of fine details, such as possible errors.

The on-line documentation contains a list of 3636 viruses which *IBM AntiVirus* claims to be able to detect. Another thousand lines of cross-reference information are provided, which permit searching for virus name through a common alias. Also included is a more detailed explanation of 153 of the more common viruses, a set which seems well chosen. Along with details of the Family/Classification of each, a paragraph explaining how the virus operates is provided.

Installation

Two different methods of installation are described in the documentation, one of which operates under DOS, one requiring *Windows*. Curiously, both methods install the files required for operating the product under *Windows*.

Installation had to be done using DOS, as the *Windows* SETUP seemed to be missing from the master disks - a bad omen? Shortly after installation commenced, the program asked whether an 'Emergency Diskette' should be made. Being cautious, I answered yes. It proved impossible: the program requested that disk 3 was inserted, then failed to recognise it correctly. I restarted, and re-installed without making an emergency diskette. No matter what I did, the program stopped after installing 29 files (750 KB), produced the wonderfully vague error message: 'Error in transferring IBM AntiVirus files', and refused to continue.

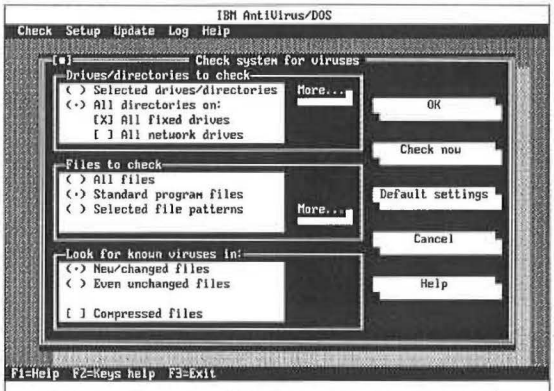
This error was at least consistent - another set of disks sent to *VB* at the same time exhibited the same problem. So here I am, for the second consecutive month with a product which would not install correctly. After a few tests, and several phone calls to the manufacturer, it was apparent that the second and third disks in the set (which seemed identical) contained files dated 1987 (I get all the most recent stuff!): they referred to mouse drivers with instructions provided in Swedish, French, German (and seemingly every conceivable European language).

The fact that the second and third disks were identical, and contained nonsense, was not the source of the problems described above. The installation process did not even get as far as asking for disk 2 before it died.

To cut a long story short, I downloaded a new version of the software from the *IBM* BBS. This worked properly, and installed under DOS and *Windows*. A plaudit is in order here for the Technical Support people, who did well in digging me out of my hole. I always received sensible advice, phone calls were returned promptly, and a solution did eventually appear. Maybe they've had a lot of practice! Just a joke...

The new downloaded version of the product gave no trouble. The DOS version installed 49 files which occupied 1.65 MB; the *Windows* version, 57 files in 2.99 MB. DOS installation takes significantly less time than that for *Windows*. *IBM AntiVirus* installed all its files into a personally selected location, and is able to alter AUTOEXEC.BAT, or store the desired changes in a separate file for later manual insertion.

Under both DOS and *Windows*, the install program offered to make an emergency diskette containing a stripped-down *IBM AntiVirus*, for use *in extremis*. I am sure that many users would infer from its name that the emergency diskette would facilitate resurrection of a PC if anything went wrong, i.e. that the floppy was more than a diskette-based virus detection system - which it is not.



Various options are available for scanning. The time taken by *IBM AntiVirus* to complete a scan, after its initial execution, compares favourably with the market leaders.

De-installation is very simple, albeit not self-evident. If the *Windows* version has been used, it is simply a matter of removing a line from the file WIN.INI, removing two lines from AUTOEXEC.BAT, and manually deleting the *Windows* group and the associated *IBM AntiVirus* icon.

Disk Checking

The first time the product executes, it says it is 'initialising its database', i.e. it searches through all hard disks to decide which files should be checked, scans each, and, if uninfected, calculates a checksum for each. This takes a long time (11 minutes 2 seconds under *Windows*, 9 minutes 59 seconds with DOS), but only happens on installation. All subsequent executions use this database to verify that files are unchanged, and scanning is then required only if something is found to be new, or altered in any way.

Many setup options are provided: an automated check (each boot, daily, weekly or monthly), checking inside compressed files (this is switched off by default, and adds considerably to the overall time taken to check a disk), scanning of high as well as 'normal' memory, and specification of any desired combinations of drives/files.

Although all options were left at their default values, the DOS version of the product detected 656 objects which required scanning, but the *Windows* version only found 648. *IBM* states that this is due to the fact that *Windows* locks certain files so they cannot be scanned. In both cases, 35 seconds was spent scanning memory and counting how many objects should be scanned (mainly the latter) every time the hard disk of my test PC was checked.

Subsequent executions of the product were much faster than the initial one. The *Windows* version checked my test PC's hard disk in 1 minute 40 seconds, when scanning for new or unchanged files. Under DOS, this took 1 minute 12 seconds. Using the 'scan unchanged files' option, the time taken rose to 7 minutes 20 seconds. This confirms the speed-up offered by the tactic of looking to see which files have changed, and scanning only those which have altered. In comparison, *Dr Solomon's AVTK* performed the same scan in 1 minute 39 seconds, and *Sophos' SWEEP* in 1 minute 34 seconds.

Accuracy

The samples used for testing are listed in the Technical Details. Of the 239 parasitic viruses, 38 were detected as definite infections, 197 as probable. Only four parasitic viruses (WinVirus_14, 8888, and two copies of Starship) went undetected. All nine boot sector viruses were detected correctly, giving an overall detection rate of 98.3%. All 500 Mutation Engine (MtE) samples were detected correctly.

Results in all sets were identical whether the DOS or the *Windows* version was used. When a ZIP file containing many MtE test samples was checked, *IBM AntiVirus* said only that the ZIP file was infected, and gave no indication of how many infected files were present.

The memory-resident component of *IBM AntiVirus*, DOS Shield, contains features allowing for various types of checking. The option shown above is active by default.

The viruses found by this product are split into 'definite' and 'probable' infections. The majority, 85%, are detected as 'probable', though they are viruses. The false positive rate was zero. As for Number_of_the_Beast, Vacsina and Yankee, some samples were detected as 'definite'; others, only 'probable'. Why? *IBM's* answer is that the product only identifies a virus as 'definite' if it is byte-for-byte identical with the one analysed; if similar, it is described as 'probable'.

Memory-resident Program

IBM AntiVirus includes a memory-resident feature called DOS Shield, comprising several components which are loaded sequentially, as desired. The separate parts claim to 'Prevent common DOS viruses', 'Warn when viral activity occurs', 'Check diskette boot records', and 'Check files when opened'. Each component provides a concise onscreen explanation of its function when it loads into memory. Only the first and third of these components are active by default; the others must be explicitly selected.

The setup screen gives an accurate indication of how much memory various combinations of these components will use. Although high memory can be used to reduce the amount of conventional RAM that is required, only one component (Prevent common DOS viruses) can use expanded memory. Use of high memory and/or expanded memory can be altered at will by the user.

When all four components are active simultaneously, 18 KB of conventional (high) memory, and 16 KB of expanded memory is required; an eminently acceptable total.

Memory-resident software is notoriously difficult to test with accuracy, but I did my best. With all the memory-resident components active, I used *Norton Commander* to copy a test-set containing one of each of the viruses listed in the Technical Details section (148 viruses in total) from one disk to another. DOS Shield reported 28 files as infected - not encouraging. *IBM's* rationale is that DOS Shield should focus on those viruses which the user is likely to encounter.

I often use 4DOS (a command interpreter which is a replacement for COMMAND.COM): when this was in use, and infected files were copied using the COPY command, all infected files were detected correctly.

After 52 files had been copied, this command produced the onscreen error message 'Too many open files' for each file it attempted to copy. After this, the PC produced that same error in response to every DOS command, and a reboot was required. If COMMAND.COM was used, COPY terminated when the first virus-infected file was encountered, and an error message appeared on screen. Version 2.3 of the product, according to *IBM*, does not contain this problem.

The memory-resident program did not detect virus-infected files within a compressed ZIP file. This is unsurprising, as such a facility would probably add a large overhead to system execution. However, when I extracted virus-infected files from a compressed ZIP file, there was no complaint from the software. Given that this created many new virus-infected files, it did seem something of an omission.

I tested the overhead added by the memory-resident software by copying 20 files (585 KB): the time taken to do this was approximately the same whether or not DOS Shield was installed, and no matter which component parts were active. Oddly, my timing measurements showed much greater variation when DOS Shield was installed. Given that the variation could be anything up to a one-second alteration in an 11 second file copying time, this was much larger than any possible measurement error which I might have made. I cannot think of any reason why this should happen.

The documentation does not explain the constraints imposed by the behaviour blocker component (it never does!). Therefore I formatted a floppy disk, ran SYS, then ran *Norton's* formatting program, and even edited absolute sectors of a floppy disk. All to no avail - I could not induce an error message. Contact with *IBM* revealed that the company has designed DOS Shield to be able to distinguish between viral and normal system activity.

The Rest

Although DOS and *Windows* versions of *IBM AntiVirus* were provided, I could detect no difference between the two, apart from some screen representation details. Even the selections available on the drop-down menus are almost identical. On my test PC it took 10.9 seconds for the DOS version of *IBM AntiVirus* to load. Given that this was a 33 MHz 486, it is likely that loading could become turgid on a slow 386, and unusable on anything less powerful.

Disinfection facilities are provided with *IBM AntiVirus*, but in common with my usual practice, I have not assessed this capability. Be safe, delete all infected files; you know it makes sense. *IBM AntiVirus* maintains three logs files whilst disks are being checked: these provide thorough details of what happened on the last execution, the previous execution, and a cumulative log of all previous checks.

Conclusions

Given the problems I had with the version of *IBM AntiVirus* originally provided for review, the phrases 'thorough testing' and 'lack of' (in no particular order) spring to mind. If *IBM* cannot come up with software which works when they know it is being provided for a review, what chance do ordinary punters have?

IBM AntiVirus detects viruses accurately and in a timely fashion. By combining the features of a scanner and a checksummer, the time taken to perform the initial check of a hard disk is quite slow. However, this only happens once, and all consequent checks are carried out more quickly than would be the case if scanning alone were used.

Indeed, using its tactic of combining a scanner and a checksummer, *IBM AntiVirus* can check disks at speeds which are faster than most anti-virus programs. Scanners which blindly search rarely-accessed corners of a hard disk are blundering through their search process for no reason, so it does seem logical to try and combine scanning and checksumming. As long as it is done carefully.

The memory-resident component is not very good at spotting virus-infected files, and does not seem to prevent a user carrying out harmful actions. However, it occupies very little memory, and does not impose a large overhead. I suppose we should be grateful for small mercies.

All this takes me back to the comparative scanner review published in the July edition of *VB*. This contained the conclusion that *IBM AntiVirus* was 'one of the slowest products tested'. I disagree. The above review has shown that this is only true the first time a disk check is invoked. On subsequent checks, *IBM AntiVirus's* combination of a scanner and a checksummer makes it faster than most products which rely solely on scanning.

Technical Details

Product: *IBM AntiVirus v2.2* (no serial number available).

Developer/Vendor (UK): *IBM UK* , Normandy House, Alencon Link, Basingstoke, Hants, RG21 1EJ. Tel 01256 314558, fax 01256 332319.

Developer/Vendor (USA): *IBM Corporation*, Long Meadow Road, Sterling Forest, NY 10979-0700. Tel +1 914 759 2901, fax +1 914 784 6054. Note also that *IBM* provides support for its *AntiVirus* program through its usual outlets in almost every country in the world. The documentation contains a voluminous list of contact addresses and telephone numbers.

Availability: Any *IBM PC*, PS/2, or 100% compatible with 640 Kbytes of RAM, and DOS version 3.3 or above.

Price: 1-250 users, £1000; 251-500, £2000; 501-1000, £4000; 1001-2000, £6500; 2001-3000, £9500; 3001-5000, £12,500; 5000+ on application only. Includes quarterly updates.

Hardware used: A 33 MHz 486 PC clone with 3.5-inch (1.44 MB) floppy disk drive, 5.25-inch (1.2 MB) floppy disk drive, a 120 MB hard disk and 4 MB of RAM, using *MS-DOS v5.00*, *Windows v3.1* and *Stacker v2*.

NB: For full details of viruses used for testing purposes, please see *VB*, May 1995, p.23.

SUBSCRIPTION RATES

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

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

Editorial enquiries, subscription enquiries, orders and payments:

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

Tel 01235 555139, International Tel +44 1235 555139
Fax 01235 531889, International Fax +44 1235 531889
Email: editorial@virusbtn.com
CompuServe address: 100070,1340

US subscriptions only:

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

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

END NOTES AND NEWS

***Compsec 95* will take place in London** from 25-27 October 1995. For details on the conference, contact Sharron Emsley at *Elsevier Advanced Technology* on Tel +44 1865 843721, fax +44 1865 843958, email s.emsley@elsevier.co.uk.

Information Security on the Internet is a **two day conference** taking place at the *Cumberland Hotel* (London, UK), on 25/26 September 1995, with post-conference workshops on 27 September. Tel +44 181 332 1112, fax +44 181 332 1191 for information.

The ***22nd Annual Computer Security Conference and Exhibition*** will be held in Washington, DC from 6-8 November 1995, under the auspices of the *Computer Security Institute* (*CSI*). The conference will feature over 120 sessions on various topics. Further information is available from the *CSI* on Tel +1 415 905 2626, fax +1 415 905 2626.

The **next round of anti-virus workshops being held by *Sophos Plc*** is scheduled for 22/23 November 1995. The two-day seminar will take place at the company's training suite in Abingdon, and costs £595 for both days (or £325 for one day only). The first day's sessions comprise an introductory course on computer viruses, and the second day is an advanced virus workshop. More information is available from Julia Line on Tel +44 1235 544028.

A **new *Macintosh* virus** has been found in the wild: HC-9507 causes unusual system behaviour, linked to the day of the week and the time: screen fade-in/fade-out, automatic entering of the word 'pickle', or system shutdown/lockup. It infects HyperCard stacks under *Apple Macs* running system 6 and 7.

IBM has announced the release of an **integrated suite of anti-virus products and services**, including software which protects PC users by detecting and removing more than 6000 strains of computer virus. The *Desktop Edition*, targeted at home users and small businesses, runs on *OS/2*, DOS, and *Windows*, with *Windows NT* and *Windows 95* support planned for late 1995. Aimed at large businesses and client/server environment, the *Enterprise Edition* includes *IBM AntiVirus for OS/2*, DOS, *Windows*, and *NetWare*. For information, contact Andrea R. Minoff at *IBM*; Tel +1 914 759 4713, email minoff@watson.ibm.com.

The ***European Security Forum Annual Congress*** will be held in Cannes, France, from 15-17 October 1995. Information on the conference can be obtained from June Chambers at the European Security Forum's London offices; Tel +44 171 213 2867, fax +44 171 213 4813.

Fischer International is about to launch a **data security product for *OS/2*, *Watchdog***. The current product line provides security for DOS and *Windows*. *Watchdog for OS/2* is now undergoing beta-testing, and will start shipping when *IBM* releases its new security hooks for *OS/2*. Further information is available from Liz Menches at *Fischer*; Tel +44 1923 859119, fax +44 1923 859151.

S&S International will be holding two rounds of ***Live Virus Workshops***; on 18/19 September and on 9/10 October 995. Cost for the two-day seminar is £680 + VAT. Further details can be obtained from *S&S International*; Tel +44 1296 318700, fax +44 1296 318777.

The *National Computer Security Association* (*NCSA*) has organised a ***Firewall Product Developers' Consortium*** (*FWPD*) to bring together the major vendors of network and Internet firewall products. According to Dr Peter Tippett, *NCSA* president, the effort is meant 'to bring together the vendors of firewall products, consumers who buy these products, and the best security experts we know'. Information on the initiative is available from Bob Bales at the *NCSA*: Tel +1 717 258 1816, fax +1 717 243 8642, email bbales@ncsa.com.