ISSN 0956-9979 APRIL 1998

VIRUS BULLETIN

THE INTERNATIONAL PUBLICATION ON COMPUTER VIRUS PREVENTION, RECOGNITION AND REMOVAL

IN THIS ISSUE:

• **Do it yourself:** Jimmy Kuo kicks off a self-help series about macro viruses. His suggestions on how to deal with macro infections without a trip to the bank start on p.9.

• **Macro progress:** A new form of *Excel* macro virus was discovered a couple of months ago and we have an analysis on p.16. Then, just before going to proof, the first macro virus for another major application in the *Microsoft Office* suite surfaced – our analysis is on p.15.

• **Give us a Clue:** Our insight column this month covers *Dr Solomon's* Senior Technology Consultant. Put a face to the name, starting on p.13.

• **Is eSafe?** This issue sees an extended product review as we put *EliaShim's eSafe Protect* through its paces. See the results on p. 20.

CONTENTS

EDITORIAL

Access all Areas

Panic! Worry. Doom. Gloom. The sky is falling…

But no, it is just a new kind of very predictable computer virus (in fact, it borders on the boring). Good headlines, though. But what was all the fuss about again? Some macro virus written in 646-bytes of Visual Basic for Applications, spanning some 23 lines of code.

Whoopdee doo! Oh yes, I almost forgot – it happened to be the first *Access* macro virus.

“ …is it, in any real sense, a ‘threat’? ”

So why the fuss? Well, it is ‘new’ and that is 75% of ‘news’. Not a good explanation. Someone was first to find it (think about it – that is the way things have to be, otherwise nothing would ever be found!). Most of what is loosely referred to as Western Society seems fascinated with ‘firsts’, which still amuses me. I always preferred getting things right, but I suspect ‘Trivial, simplistic virus will be no threat so do not be concerned’ doesn’t quite have the ring that a ‘first’ has.

A97M/AccessiV is what is known as a ‘proof of concept’. Wow! So you (or someone) can write a virus in *Microsoft Access.* Anyone who understands anything about PCs has known for some time that *Access 97* (and quite likely *Access v2.0* and up) had sufficient power to support viruses in its macro language.

DOS batch language would never win prizes for its complexity, but it supports viruses too. Has the world come to an end? DOS was (hell – still is!) at the heart of most day-to-day computing, so viruses attacking something as fundamental as DOS batch surely must be ‘bad’ (a better option than ‘first’ if you are a journalist and have a surfeit of stories), mustn’t they? Maybe DOS batch viruses did put an end to modern computing and we just failed to notice?

Someone has written a virus in *Access*, but I’m sorry; I just cannot get terribly excited about it. The important question is, is it, in any real sense, a ‘threat’? You will not be surprised that I think not. Read about the rapid rise of *mIRC* script viruses on p.7 of this issue. Notice that IRC continues largely unaffected and that *mIRC* software is still one of the most popular IRC clients. Notice also that *mIRC* viruses are not a big problem a mere three months after they became a ‘news item’. The user communinty and developers have worked through the solutions. Yes, there will be *mIRC* viruses as long as two people on IRC either run an old version of that client or misconfigured versions, but at present it is really just a mopping-up exercise.

Another interesting virus fact – the ever-popular PostScript display language supports viruses. Proof of concept examples were written ages ago; maybe before the first copy of any version of *Word* appeared. Why have people kept buying such dangerous printers all this time?

Am I getting too old and cynical? The first *Access* virus really is ‘news’, isn’t it? Nope. Those trying to make it big news have some other agenda. Me – I’d be more worried about the half-dozen or so trivial variants of Wazzu that *Microsoft Word* itself has probably created in the last 24 hours. One of them has a modest chance of spreading quite some way undetected. Together (probably singly, actually) they will cause more damage and cost more productive time to be lost than this new *Access* virus.

…except that the anti-virus industry has probably already expended several hundred thousand dollars in time (analysing it, beginning to reverse engineer the *Access* file format so they can claim to be the first to disinfect it properly, and in press releases and promotion).

You should be worried (though not in the ‘shock, horror’ way) about new viruses we are seeing developed that work properly under *Windows 95* and *NT*, or some of the sophistication we are seeing forced on macro virus writers by changes in *Word*, or large complex viruses that include possibly 20 KB of networking code that no-one has time to disassemble and untangle. But ‘Virus writer in portable KERNEL32 GetFileAttributesA shocker’ doesn’t quite seem like news…

NEWS

The Name Game

Recently, *KAMI Corporation's* anti-virus department was officially renamed *Kaspersky Lab* after head *AVP* developer Eugene Kaspersky. Following the break-up of *STC KAMI Corp* into several independent companies, *Kaspersky Lab* is now the sole owner of the *AntiViral Toolkit Pro* range of anti-virus products.

With a fast-growing team of twenty-five employees, and the successful launch of *AVP* v3.0, *Kaspersky Lab* aims to become the number one anti-virus company in the Russian market. Natalya Kasperskaya, CEO of *Kaspersky Lab*, is optimistic that the organization's new, independent status will allow the *AVP* team to concentrate on research and development. Additional information about the reorganization of *Kaspersky Lab* is available on the World Wide Web at http://www.avp.ru/english/ ∎

EICAR '98 Conference

This year's EICAR (European Institute for Computer Anti-Virus Research) Conference was held in Munich, Germany from 16–18 March. The focus of EICAR has broadened recently and, in fact, the subject of Web Safety was billed as the primary topic of the conference. Although *Virus Bulletin* was not represented directly, we had our moles checking out proceedings.

Steve Branigan, a JAVA expert from *Lucent Technologies*, showed the bright lights, but also the darker aspects, of JAVA applets. Cryptography, privacy and web security issues were broadly covered during the three days. An interesting panel session with Sarah Gordon, Marko Helenius, Alan Solomon and Steve Trilling about non-viral malware and the anti-virus industry's response to it, was well-attended at the end of the last day.

Macro viruses and the *Windows* environment have caused complex challenges for those evaluating anti-virus products. The number of macro viruses is growing all the time and the user interface creates problems for several tasks required in automated product testing. Marko Helenius (University of Tampere, Finland) presented some of his experiences with these issues. In a related theme, there was a general consensus among the anti-virus experts present that the macro virus problem should again be addressed by *Microsoft*. If nothing else, the hope was that this would support anti-virus software producers in the battle against macro viruses.

Attendees travelled from all parts of the globe and all corners of the industry to attend the conference. There were representatives from *Dr. Solomon's Software* (Alan Solomon, Dmitry Gryaznov, Igor Muttik), *FRISK Software International* (Vesselin Bontchev), *Data Fellows* (Mikko

Prevalence Table – March 1998

Virus	Type	Incidents	Reports
Cap	Macro	99	22.3%
AntiExe	Boot	35	7.9%
Form	Boot	26	5.9%
Concept	Macro	25	5.6%
Parity_Boot	Boot	23	5.2%
Dodgy	Boot	13	2.9%
Wazzu	Macro	13	2.9%
Laroux	Macro	11	2.5%
Npad	Macro	11	2.5%
NYB	Boot	11	2.5%
Empire_Monkey	Boot	10	2.3%
Impost	Macro	10	2.3%
AntiCMOS	Boot	9	2.0%
Ripper	Boot	9	2.0%
Sampo	Boot	8	1.8%
Angelina	Boot	7	1.6%
DelCMOS	Boot	5	1.1%
Junkie.1027	Multipartite	5	1.1%
WelcomB	Boot	5	1.1%
Appder	Macro	4	0.9%
Junkie.1000	Multipartite	4	0.9%
Quandary	Boot	4	0.9%
Temple	Macro	4	0.9%
ABCD	Boot	3	0.7%
Eco	Boot	3	0.7%
Edwin	Boot	3	0.7%
IntC	Boot	3	0.7%
LBB_Stealth	File	3	0.7%
Munch	Macro	3	0.7%
NiceDay	Macro	3	0.7%
Rapi	Macro	3	0.7%
Schumann	Macro	3	0.7%
Showoff	Macro	3	0.7%
Spanska.4250	File	3	0.7%
Others [1]		60	13.5%
Total		444	100%

[1] Comprising two reports each of: Anxiety, Baboon, Cascade, Exebug.G, Galicia.800, Jerusalem.1363, Natas.4738, Niknat, One_Half.3544, Satria, Stealth_Boot and Urkel; and single reports of: ABC, Anak, Barrotes.1310, Bleah.D, Boza.D, Breasts, Colors, Counter, Divina, HLL.4023, Hunter, Influenza, Johnny, Jumper.B, Junkie.1210, Kompu, Lilith, Lunch, Manic.2143, Maverick.2048, MDMA, MtE.Dedicated.G, Nightfall.5767, Russian_Flag, Screaming_Fist.711, Spanska.1000, Stoned.Kenya, Stoned.Spirit, Swiss_Boot, Swlabs, TPVO.3783, Twno.A:Tw, Unashamed.B, Vacsina-05, Vbasic.5120 and V-Sign.

VIRUS BULLETIN **©1998 Virus Bulletin Ltd, The Pentagon, Abingdon, Oxfordshire, OX14 3YP, England. Tel +44 1235 555139. /98/$0.00+2.50**

Hypponen), *IBM* (Richard Ford, Sarah Gordon), *Network Associates* (Francois Paget) and *Symantec* (Steve Trilling, Rene Visser, Motoaki Yamamura).

As usual, there was much productive work which took place after hours in the designated research areas (some of which served a variety of beverages). At one such session, a number of vendors' technical support departments were contacted to compare the service. Our man at EICAR put in the call to *Dr. Solomon's*, and with Alan Solomon sitting beside him, had the following conversation with the technical support representative:

VB Mole: Is there really a guy named Dr Solomon?
Tech rep: Yes sir, there is.
VB Mole: Well, does he still come into the office?
Tech rep: Yes, he does.
VB Mole: Does he ever do anything useful?
Tech rep: No, definitely nothing useful.
VB Mole: Hang on, someone here wants to say 'Hi'!

Too bad Peter Norton was not there to play the same game with *Symantec*… ▮

Microsoft Strikes Back

VB has become aware of a move by *Microsoft* to implement an anti-virus support interface in the next version of their *Office* suite. The proposal is for a COM (Common Object Module) interface, called IOfficeAntiVirus. Anti-virus developers should be able to implement support for this quite easily in their existing on-access scanners.

The stated goal in the *Microsoft*-sourced document *VB* has seen, is 'to empower anti-virus ISVs with the ability to design and implement anti-virus functionality that can be utilized by all the *Office 9* applications'. The proposed scheme allows for support of mulitple anti-virus products. If more than one application is registered to use this interface, each is to be called in turn. This should allow your scanner(s) to check and clean documents before they are properly opened for use by the *Microsoft* application.

The current proposal is that registered anti-virus components should open the target file, perform their tasks (which can include providing user feedback) then return control to the *Office* application. It is to be hoped that this scheme is extended slightly, so as to provide a marked improvement in user protection. *VB* recommends that the calling *Office* application should first check the target file for encryption, and pass the file to the anti-virus components as a pre-processed stream, thus allowing *Microsoft* a degree of freedom in implementing encryption (and other features, like compression) without impacting anti-virus products' ability to scan *Office* files thoroughly by these means.

Vendors will have to implement their anti-virus interface as a standard ActiveX component registered as an in-process server. This component should be registered as supporting the MSOfficeAntiVirus category. ▮

VIRUS BULLETIN **©1998 Virus Bulletin Ltd, The Pentagon, Abingdon, Oxfordshire, OX14 3YP, England. Tel +44 1235 555139. /98/$0.00+2.50**

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 16 March 1998. 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
- **D** Infects DOS Boot Sector (logical sector 0 on disk)
- **E** Infects EXE files
- **L** Link virus
- **M** Infects Master Boot Sector (Track 0, Head 0, Sector 1)
- **N** Not memory-resident
- **P** Companion virus
- **R** Memory-resident after infection

Abbas.1313

CER: An appending, 1313-byte virus containing the encrypted texts 'IRANIAN VIRUS W.by ABBAS KUHKAN ALIABADI' and 'ANTI VIRUS Writen By ABBAS KUHKAN Virus Found !'.

```
Abbas.1313          268B 0E86 008E C126 817F 034B 5574 3B2E 898D 0C00 2E89 9D0A
```

Ciudad.590

ER: A 590-byte appender containing the encrypted text '[CIUDAD REAL]VIRUS', displayed on 19 June. Infected files have the word 07CCh at offset 0012h and 4753h ('SG') at the end of their code.

```
Ciudad.590          B440 B94E 0290 33D2 CD21 2EA1 CD00 538B D8B1 04D3 EB83 C338
```

Dbell.558

CN: An appending, 558-byte direct infector, which infects two files at a time. It contains the texts '[The Division Bell] (c) 19xx by Midnight Lamp / Philadelpia, PA, USA', 'C:\DOS', 'EDIT.COM', '..' and '*.COM'.

```
Dbell.558           E8C2 00B9 2E02 B440 8D96 0301 CD21 E886 00FE 8671 0380 BE71
```

Estier.2126

CER: An encrypted, appending, 2126-byte virus containing the texts 'CAMILA V2.0 - Paraguay', 'TBDRV', 'TBAVTBSCANNAVVSAFEF-PROTSCAN' and 'COMEXE'. The payload, triggering on 17 and 28 July and 14 October, displays the first message and tries to overwrite 546 sectors on drive C.

```
Estier.2126         B9EE 07BD 1801 E800 005E 81EE 0901 2E80 32?? E304 4945 EBF6
```

HLLO.40932

EN: An overwriting, 40932-byte, fast, direct infector containing the texts 'Julius Caesar', 'by: justin', 'St. Michaels College School (ONT CAN)', 'Caesar', 'attrib +h Caesar', 'attrib -r -h -s io.sys', 'The soothsayer was correct, Caesar has been murdered!!!!', 'I wish I knew ASM', 'The ides of march have come!' and 'Error in EXE'. The virus spreads in compressed form (42,448 bytes after unpacking), and creates a hidden file called 'CEASAR'.

```
HLLO.40932          578B 3EFE 03BE 5606 ACAA 0AC0 75FA 5FE8 E50B 8BD7 B441 CD21
                    720D B44F CD21 73E0 075F 5E5D CA02 003D 0500 7503 E9D0 2AE9
```

HLLP.9840

ER: A prepending, 9840-byte virus containing the texts 'AUTOEXEC.BAT', '*.exe', 'RmLg', 'c:\nuevo.exe', 'DESTRUCTION !!!', 'GUATON', 'HOLA', 'COMSPEC' and 'c:\file111.chk'. The virus infects files when the default directory is changed.

```
HLLP.9840           E8D1 F580 3E9E 0103 751E BFA2 001E 57BF CF0A 0E57 9A8D 0970
                    01BF A200 1E57 BFDE 0A0E 579A 550B 7001 E88B F3BF BA02 1E57
```

Iblis.252

CN: An appending, 252-byte direct infector, infecting one file at a time. It contains the texts '*.com' and 'iblis [DJ Freedom]'. Infected files have the byte FEh at offset 0003h.

```
Iblis.252           B440 B9FC 00BA 3601 03D5 CD21 B800 42B9 0000 BA00 00CD 213E
```

IVP.597

CN: An encrypted, appending, 597-byte, fast, direct infector containing the texts 'XA-XA 3.0', 'A.L.S.', and '*.com'.

```
IVP.597             B928 0290 2E8A B657 0390 2E8A 2790 32E6 902E 8827 9043 90E2
```

IVP.687

CEN: An encrypted, appending, 687-byte, fast, direct infector containing the texts 'THE REAL! EXTINCTOR...', '.oO PHAETHON Oo.', '[IVP]', '*.com' and '*.exe'.

```
IVP.687             8D9E 0F01 B98A 022E 8AB6 B103 2E8A 2732 E62E 8827 43E2 F5C3
```

IVP.967

CEN: An encrypted, appending, 967-byte, fast, direct infector containing the texts 'April.Bad.Fridays[19M94]System Error! Found the: Bad Friday(s) virus! Running self diagnostics, Please wait...', '*.com' and '*.exe'. The payload, triggering on Fridays in April, displays the above message and tries to overwrite 255 sectors on every logical drive from A to Z. Infected files have the word 6D6Dh ('mm') at offset 0003h (COM) and at offset 0010h (EXE).

```
IVP.967             8D9E 1501 B9A0 032E 8A27 2E32 A6CB 042E 8827 43E2 F2C3 ??
```

Pempe.1943

ER: An encrypted, appending, 1943-byte virus containing the texts '*.EXE', 'P E M P E V1.6' and 'AMACC (Makati,Phils) [PM]'. Infected files have the word 4D50h ('PM') at offset 0012h.

```
Pempe.1943          26A1 8400 501E 07B8 DEDE CF3D 4D50 7502 EB2F BF00 10BE 8811
```

Roet.1876

CER: An appending, 1876-byte virus containing the texts 'ROETVIRUS_E5IB INFECTION PC DeStruCTion aFTeR 255 bEEpsTry aGAin NexT YEar' and 'ROETJE - THE IMMENSE'. The payload, triggering between 20 and 31 December, tries to overwrite 6000 sectors on drive C.

```
Roet.1876          B888 B1CD 21EB C374 2F8C C0EB C481 2E12 0080 00EB C88B F5B9
```

Sphinx

CN: Polymorphic, appending, direct infectors containing the texts '*.c*' and '????????C??'. Infected files start with the byte AEh. A 2534-byte variant also contains the text 'SPHiNX v1.0' and a 2548-byte variant contains '[SPHiNX]'. It is impossible to detect infected files using a simple template.

Trout.6804

CR: A polymorphic, appending virus with a constant 6804-byte encrypted code but widely variable polymorphic decryption procedure. The difference between particular infections can be greater than 1000 bytes. The virus contains the texts '2TThis is "The Second NewBorn Trout" virus Programmed in the city of Milan, North Italy [C] The Tricky Trout 1995 Mutation Engine: TT-PEB (Tricky Trout Plurimorphic Encryptor Builder) version 2.01 (TPE standard)' and 'SCAN.NETSCAN.CLEAN. VSHIELD.F-PROT.VSAFE.MSAV.CPAV.NAV.TBAV.TBSCAN.VDS.NOVI.AVP.-V.-VPRO.VIREX.AVSCAN. VI-SPY.GUARD.FINDVIRU.TNT.TNTAV.HTSCAN.NEMESIS.NOD.ITAV.VI.VIRIT.IM.WIN.TD.DEBUG.', 'CHKLIST.MS', 'CHKLIST.CPS', 'ANTI-VIR.DAT', '\SCANCRC.CRC', '_CHK.CHK', '\NAV_._NO', 'C:*.COM', 'C:\DOS\KEYB.COM', 'C:\DOS*.COM' and '*.COM'. Infected files contain the word 5432h ('2T') at offset 0003h. The following template may be used for detection in memory only.

```
Trout.6804         B853 63B1 00CD 213D 5432 8CC8 8ED8 C31E 2BC0 8ED8 C706 0C02
```

Vanitas.2040

ER: An encrypted, 2040-byte appender containing the texts 'C:\COMMAND.COM', 'VANITAS++ v1.1 (c)GR97 by ANAX.Member of <VBB>.Everyone is Original. [E-75]Chill-Out.' and 'C:\WINDOWS\COMMAND.COM'.

```
Vanitas.2040       BB0E 0143 43B4 02CD 162E 8107 ???? EB00
```

VCL.558

CN: An encrypted, appending, 558-byte direct infector, infecting two files at a time. It contains the texts '*.*', '*.com' and '[VCL]'. The virus switches between two possible encryption procedures.

```
VCL.558            8DBE 0E01 B909 0181 35?? ??47 47E2 F8C3
VCL.558            8DB6 0E01 B909 0181 34?? ??46 46E2 F8C3
```

VCL.870

CN: An appending, 870-byte direct infector containing the texts 'Xt-FUCKED YOUR HARD DRIVE! YOU'LL GET OVER IT!', 'Size differece for virus here is the date of something special that happened to me.', '*.*', '*.COM', '[SVI]', 'Virus name IS... SCHAINE... Spell it right ppl.', 'Schaine is the best. Do you understand that?', 'That is what my favourite person calls me. k.' and 'This is my first. My next will be better...'.

```
VCL.870            B440 B966 038D 9503 01CD 21B8 0157 8B4C 168B 5418 CD21 B43E
```

Vicodin.1262

ER: An appending, 1262-byte virus containing the texts 'Crash.Poppy v.1c' and 'by VicodinES'. The virus infects files with extensions 'ex?' and 'dl?' and avoids infecting files with names ending with 'pt', 'an', 'vp', 'av' and '32'. Infected files have the word AFAFh at offset 0012h.

```
Vicodin.1262       FCB8 FF51 CD21 3D51 FF74 102A DBCD 168C D848 8ED8 2BFF 803D
```

Vlad.Dir.757

ER: An appending, 757-byte virus containing the text '[VLAD-DIR v 2.0] [Darkman/VLAD]'. The virus infects files on Find First/Next function (e.g. during the 'dir' command). Infected files have the word 3256h ('V2') at offset 0012h.

```
Vlad.Dir.757       C744 1256 32B4 40B9 F502 99E8 1A00 B800 4299 8BCA E811 00B4
```

VME.2862

CN: An encrypted, 2862-byte prepender dropping the Hal virus (see *VB*, February 1998, p.4). It contains the text 'HAL 3001*.com'. Infected files have their time-stamps set to 58 seconds. It is impossible to detect all replicants using a simple template.

Vulcan.227/294

CR: Two simple prependers. Their 'Are you there?' call Int 21h AH=6Dh returns the value AL=93h.

```
Vulcan.227         B46D CD21 3C93 7426 B821 35CD 2189 1ED8 018C 06DA 01BE 0001
Vulcan.294         B46D CD21 3C93 7450 B821 35CD 2189 1E1B 028C 061D 020E 5848
```

Vulcan.307

CR: A 307-byte variant of the preceding containing the text 'Vulcan'.

```
Vulcan.307         B46D CD21 3C93 745E B821 35CD 2189 1E21 028C 0623 020E 5848
```

Vulcan.480/489

CER: Two viruses which prepend COM files and append EXE files, containing the text 'Vulcan'. The 'Are you there?' call Int 21h AH=6Dh returns the value AL=93h.

```
Vulcan.480         B46D CD21 3C93 C306 B821 35CD 2189 1EBE 018C 06C0 0158 488E
Vulcan.489         B46D CD21 3C93 C3FC 06B8 2135 CD21 891E BE01 8C06 C001 5848;
```

Vulcan.484

CER: A prepending (COM) and appending (EXE), 484-byte virus. The 'Are you there?' call Int 21h AH=6Dh returns the value AL=93h. Infected files have their time-stamps set to 30 or 62 seconds.

```
Vulcan.484         B46D CD21 3C93 C3FC 0606 B821 35CD 2189 1EBE 018C 06C0 0158
```

Whimsy.256

CN: An encrypted, appending, 256-byte, slow, direct infector, infecting one file at a time. It contains the text '*WHIMSY*.CO?'. Infected files have their time-stamps set to 62 seconds. Because of its slightly polymorphic encryption, it is impossible to select one simple template.

```
Whimsy.256         B501 CD21 BE5A FFB9 CF00 4E80 34?? E2FA C3E9
Whimsy.256         B501 CD21 B9CF 00BE 5AFF 4E80 34?? E2FA C3E9
```

FEATURE

Murky Waters

Snorre Fagerland
Norman Data Defense Systems

The Internet Relay Chat started as a multi-user chat system on a BBS in Finland in 1988, but quickly spread to the Internet. It is based on the notion of virtual rooms (called 'channels') where people can chat about anything they wish. While it is a very addictive system – you meet interesting people from other places and cultures – it is a fact of life you come across ignoramuses too. Still, it is easy to find folk with interests similar to yours, as there are no restrictions on the creation of new channels. If your particular fancy is collecting insects (yes, I do it too), just join the #insects channel and wait…

Enter the mIRC

In 1993, Oikarinen and Reed published RFC1459 which is now the official standard IRC protocol. For a long time before that IRC was mostly used on Unix systems running ircII clients. While this worked well (and is still in widespread use), it was hardly the most user-friendly system. With the availability of TCP/IP for *Windows*, a number of *Windows*-based clients were developed, namely *Xircon*, *pIRCh* and *WSIRC*. *OpenChat/2* came along for *OS/2*, and there were *Ircle*, *Homer*, *MacIRC* and *ChatNet* clients for *MacOS*. By far the most popular and widespread client is *mIRC* (pronounced murk) for *Windows* and *Windows 95*. Written by Khaled Mardam-Bey, *mIRC* is highly configurable, and supports the creation of scripts to automate a number of actions. Everything from sending a 'hello' message when someone joins a channel to sophisticated 'bot' functions can be accomplished via these scripts.

Script-based Viruses

Until recently, script viruses, while not unknown, have not been a great threat. A script is a set of commands intended for interpretation and execution by a compiled program, or 'operating environment'. The collection of a number of operating environment commands into a package that is replicative in itself has been implemented on the MS-DOS platform with BAT viruses – the command shell being the relevant operating environment.

Macro viruses can be considered to be script viruses – a *Word* macro is a set of commands to the operating environment (in this case *Word)*. In this light, we see that script viruses can be hugely 'successful', provided that exchange of scripts is high. Unix, *VMS* and *VM/CMS* now have script viruses, but they are not much of a threat due to higher degrees of security, comparatively limited OS installations, and their own conspicuousness.

mIRC supports a feature called Direct Client-to-Client connection (or DCC for short). DCC is not a part of IRC itself, instead it is a protocol for direct communication between users, bypassing the IRC server. In DCC you may chat privately, but also send and receive files. In autumn 1997, some *mIRC* users noticed that they were unknowingly sending a file called SCRIPT.INI around via DCC. A new kind of script virus was the culprit.

Much discussion has centred around how these scripts should be classified. Standard DOS-based viruses propagate in the DOS environment and use DOS executables as hosts. It is my opinion that we must consider IRC as the environment and every *mIRC* client as the possible host. In this context, it is plain to me that SCRIPT.INIs are indeed viruses, as they fulfill all other criteria (replication, host dependence…).

Infection Mechanism

SCRIPT.INI is a file that contains script commands, which *mIRC* will load on startup if it finds it in its home directory. Infectious SCRIPT.INIs contain instructions to send themselves on to others on specific events – typically when someone joins the same channel as the infected user.

It was discovered that current *mIRC* versions would place SCRIPT.INI in the *mIRC* home directory if it was received over DCC from another user. If people had DCC autoget enabled (automatically accept and receive file send requests), they would receive minimal warning when they were sent anything, and from their next *mIRC* session they too would be infected and sending out SCRIPT.INIs. With DCC autoget off, a dialog box pops up asking if they want to receive a file called SCRIPT.INI from user so-and-so.

A typical DCC get dialog box when someone tries to send a SCRIPT.INI (name and account details changed).

People infected with a SCRIPT.INI virus are thus very conspicuous and, once the problem was widely known and understood, would usually receive many less than polite notes about where they should put their scripts.

Infection Rates

So how infective are these beasts? Not very. I would guess (much of this is assumption, based on what I have personally witnessed) that in mid-January 1998 the infection mass was about 1% of IRC users and the infection rate in a

channel with one infected user was about 10%. Infected users are usually kicked off a channel quickly or 'persuaded' to delete their SCRIPT.INI file. This explains the discrepancy between infection rate and infection mass. The best conditions for spread are the heavily-populated general chat channels, like #chatzone. The more users on a channel, the greater the chance that one of them will be infected, and that more people will join and leave. It is not unusual to find channels with more than 400 users at a time, bombarding you with four or five DCC send requests when you join.

What to Look Out For

These viruses are rarely more than an annoyance, making you unpopular among other IRC users by sending your INI file around uninvited. The potential for much greater damage is there though. Most 'ordinary' destructive viruses today carry with them a non-specific payload, designed to destroy data belonging to the unlucky infectee. *mIRC* scripts carry with them a multitude of payloads, some quite innocent, others very insidious.

A number of script viruses, particularly those belonging to the Jeepwarz, Acoragil and Whacked families, redirect whatever you type on the keyboard to other channels on IRC. Many channels are set to invite-only, meaning that members can have them to themselves, without anyone else joining. IRC was designed to allow for private conversation through the /msg (private message) system, not to mention DCC chat, which is even more private as it does not go through the IRC server network at all. Of course, this enables many people on IRC to discuss things that are private in nature (yes, I mean sex). Unfortunately, if their systems are infected, they might be entertaining many more readers with their stories.

mIRC supports the /fserve command, which sets your PC up as a fileserver. This means that others can perform DIR, GET and PUT commands remotely, much like FTP. It is possible to use this command in a script as well and several of the SCRIPT.INI viruses use it. Also, many scripts connect different commands with file send actions, such as initiating a DCC send c:\linux\etc\passwd when a certain command is used either publicly on the channel or in a private message. This basically means that if you are infected, all of your files are world readable.

As if global read rights are not bad enough, a number of commands allow a script to perform write actions on the infected system. The command remini is used by the script mIRC/RemIni.2035 to blank large sections in the Windows WIN.INI and SYSTEM.INI files, rendering the system practically useless. The Acoragil family uses the command writeini to create a PROTECT.INI file. This contains information about which particular IRC users are to be protected if the infected user has operator status.

The script even has execute rights, leaving the system totally vulnerable. Two script families currently use the /run or .run command. mIRC/Play uses the run command to call ATTRIB.EXE, setting the script itself to read only. mIRC/Whacked – potentially one of the most dangerous scripts – defines an alias such that if anyone types 'runthis anyfile.exe' either publicly on the channel or in a private message, the infected system will run ANYFILE.EXE. The implications of this should be obvious.

The commonest destructive actions of these scripts are less serious. Simpsalapim, Acoragil, Cure, Gakk, Panterax and others cause the user simply to quit IRC on a certain keyword. Also, they very often implement keywords as infection notification triggers. Simpsalapim, for example, will send the message 'i'm iNFeCTeD!' to anyone writing 'XX!XX' on the channel.

This latter feature has also been used to rid channels of infected users, since it makes them easy to spot. Thus, it should not be surprising that perhaps the most 'successful' script, mIRC/Jeepwarz, does not have a 'self-identification' feature like this. Many scripts randomly transmit various text messages. One script, Pedro, causes the afflicted user to join the #gaybrasil channel if someone types '!!join' on their current channel. Some people might consider this a favour, but many would not.

Combatting Script Payloads

The most obvious of the negative script payloads, the redirection of messages, is being handled quite well on the different IRC networks. On EFNet and Undernet, according to some inside sources, voluntary IRC operators are locking down most of the channels used for listening in on messages from infected users. On DALNet the omnipresent and omnipotent automaton ChanServ monitors those channels known to receive redirected messages. You are allowed to join the channel, but ChanServ will appear almost immediately and kick you out, as in this example, where names have been changed:

```
Session Start: Sun Mar 08 14:57:44 1998
*** Now talking in #jeepwarz
<Nick01> (=Nick02) what is that..?
<Nick03> (#poly) bye bla
*** ChanServ changes topic to
'closed!!!!!!!!!!!!!!!!! (mittens)'
Session Close: Sun Mar 08 14:57:45 1998
```

As you can see, within a second the #jeepwarz channel receives redirected messages from two infected users. The first (Nick01) is typing in a private DCC chat connection, while the other (Nick03) types openly on the #poly channel. The authors of the scripts keep creating new redirection channels, but the commonest have been closed.

It is difficult to estimate the number of machines that have been set up as fileservers or compromised in other ways (hacks due to strangers obtaining /etc/passwd or different destructive actions). I do not believe that these back doors are being used to any great extent, but I do not have any figures to back up this claim. In any case, these payloads are impossible to stop without removing the source of the problem, the script itself.

Combatting Scripts

Fortunately, at the client level it is easy to combat *mIRC* script viruses. All you have to do is delete the SCRIPT.INI file, and set the DCC On Send Request option to 'Show get dialog' or 'Ignore'. Then you should be quite safe in future. Combatting scripts across an on-line user mass of perhaps 25–50,000 is far more difficult.

By far the most important step towards ridding the world of these scripts was taken by the *mIRC* authors themselves. Since version 5.30 the *mIRC* home directory has no longer been the default download directory, making it difficult for an unsuspecting user to be infected. As more users upgrade their *mIRC* clients, SCRIPT.INI viruses will slowly die out. You can easily check which version of *mIRC* you are using from the first line in the program's splash – the example here is from the current release, v5.31.

In Norway, many of the people on the popular channel #norge in the daytime are students, on-line from IRC clients on machines in their school or university. These students, as a rule, know little about computers and even less about security. Further, maintaining school computers is not their responsibility. They leave the maintenance to the teachers, who know more about computers, but who often are overworked and not very up to date. The result is that the IRC clients are not upgraded, the problem goes unnoticed, and Norwegian schools continue to infect both each other and foreign schools indefinitely.

Some steps are taken by the on-line community itself. On the bigger IRC networks, users on the most popular channels regularly issue 'magic words' that will either cause infected users to quit IRC or signal their presence. Infected users are kicked off the channel with a short message about getting rid of the script. The awareness about the problem is increasing, but slowly, because new IRC users join every day.

So, what can anti-virus companies do about this problem? In my opinion, it is not strictly necessary to include these viruses as targets in anti-virus products. That is not to say it is a bad idea – if you have the time, fine, add them in. The problem is going to go away whatever the anti-virus industry does, and, to my knowledge, there has been little data loss in connection with this virus group.

Some helpful URLs

IRC protocol: http://ds.internic.net/rfc/rfc1459.txt
DCC protocol: http://www2.undernet.org:8080/~cs93jtl/irc_dcc.txt
Script.ini help: http://www.irchelp.org/irchelp/mirc/si.html

TUTORIAL

Free Macro Anti-virus Techniques

Jimmy Kuo
Network Associates Inc.

[Jimmy is Director of Antivirus Research at Network Associates. *Over the next few issues of* VB *we will feature a serialization of the paper he presented at VB'97. His well-received paper discussed the pros, cons, whys and wherefores of many free macro anti-virus techniques. These include* Word's *own 'Prompt to Save Normal.dot', making your NORMAL.DOT read-only, using the shift keys, disabling automacros, keeping a backup of NORMAL.DOT and checking against it, as well as methods to counter* Excel *viruses. Some of these simple procedures are widely touted in various Internet discussion groups, but Jimmy shows that none alone is a macro virus panacea. In the final installment of this series, Jimmy will describe the methods he uses to protect himself against macro viruses in his day-to-day work. Please note that screen shots, menu and command names and file locations are (unless otherwise stated) from a default installation of* Word 95 v7.0a *and may differ slightly for* Word 6 *and* Word 97 *users. Ed.*]

It is more than two years since Concept was unleashed upon an unsuspecting world. In that time, *Word* macro viruses have become the most prevalent virus threats to organizations. Multitudinous 'buy my macro antivirus' solutions are available, but let us start with things you can do for free.

Introduction to Macro Viruses

Any product affording its users the capability of writing macro code that in turn can write to the disk and propagate more macros, can sustain macro viruses. The platform with the most macro viruses at present is *Microsoft's Word for Windows*. Viruses spread easily in this environment because documents can contain both text and macros. However, by combining text and macros, the user has much more power and thus, better usability features. The two go hand in hand – more power to the user means more potential for macro viruses. As it is *Microsoft's* intention to achieve the former, we will all have to contend with macro viruses for a long time to come.

Excel, also from *Microsoft*, is similarly afflicted. Laroux, the first *Excel* macro virus, followed the first widespread *Word* macro virus, Concept. The same dynamics that affect *Word* also affect *Excel*. *Excel* also uses *OLE2* container files which have macros and all of the cell functions and data in the same file. When *Office 97* came along, all the macro languages converged on Visual Basic for Applications 5 (VBA5), making cross-application viruses a theoretical

possibility. This makes the possibility of macro viruses for other application platforms only a matter of time, as more and more vendors independently support VBA5.

Word macro viruses replicate most easily if they intercept the macro execution path by using one or more 'auto' macros, or by using a menu replacement. They typically enhance their chances by copying themselves into the global environment (usually by updating NORMAL.DOT). Most of the following techniques target that scenario.

In the Northern summer of 1995, *Microsoft* unleashed Concept.A upon the world. The world, anti-virus companies, and *Microsoft* itself, were not prepared for this. *Microsoft* came up with the following suggestions.

Prompt to Save Normal Template

Prompt to Save Normal Template is a *Word* configuration option. It became the first macro virus prevention method when *Microsoft* suggested its use against Concept.A. To activate the option, click on Tools then Options, and choose the Save tab. About the middle, on the left, check Prompt to Save Normal Template.

As mentioned above, a virus must spread. This is most easily accomplished by its getting into the global environment, represented by the NORMAL.DOT file. If a virus attempts to alter NORMAL.DOT and this option is enabled (it is off by default), *Word* will inform you that there is a request to change the Normal Template as you attempt to exit. You can respond that you do not wish to allow such a change. The user would know of any intentional changes and, presumably, know the right answer.

However, even with this option in use, it is still possible to open an infected file and infect the environment. The warning does not occur until exit from *Word*. Thus, any documents opened and saved after the initial infected document will also be infected.

The benefits derived from setting Prompt to Save Normal Template are similar to those of making NORMAL.DOT readonly (discussed in detail below). The primary difference between the two is that Prompt to Save Normal Template is easily turned off from inside *Word* and many viruses have already incorporated this 'defence'. As an action from within *Word*, it has the advantage of being activated prior to attempts to write to NORMAL.DOT. No law prevents the use of both, however! Since they are both free and do not conflict, why not?

Pro: Easy to set.

Commands required to set this feature can be automated.

Protects against some of the commonest in the wild viruses, like Concept.A, Wazzu.A, and NPad.

Con: Easy for a virus to turn off 'invisibly'.

Does not inform until *after* infection. All documents accessed after the infected document in that session can be infected.

Any benefit achieved is a subset of the ReadOnly NORMAL.DOT. section.

Create a Payload Macro

The second suggestion from *Microsoft* was to create a macro called Payload. I will not dignify this any further by explaining how to implement the idea.

Pro: Protects against Concept.A.

Con: Does little else.

This macro can be captured by viruses (see below).

Install ScanProt

ScanProt shows that those unfamiliar with the anti-virus industry are doomed to repeat many of the mistakes that were made in its infancy. *ScanProt* targets just one virus. The anti-virus industry was born of similar methods, but soon learned this was not a viable, long-term solution.

A macro package written by *Microsoft*, *ScanProt's* original intentions were to protect against Concept.A and provide an alert mechanism for users opening documents containing macros. These noble objectives have since been incorporated directly into the product in versions 7.0a and *Word 97*. Another *ScanProt* action is to rename macros associated with some viruses. Thus, it renders viruses non-functional, and irremovable by some anti-virus products.

Also unintentional is the misfortune whereby various *ScanProt* macros are 'absorbed' into other viruses. This causes a 'mating' scenario between an existing virus and *ScanProt* and unfortunately produces a virus variant. Anti-virus developers endeavour to solve user problems without creating new ones. This scenario makes *ScanProt* something we cannot recommend, even with its virtues.

Pro: Protects against Concept.A.

Alerts if any macros exist before a document is properly opened.

Con: *ScanProt* macros have been absorbed into many viruses and now spread as part of the virus.

Can prevent proper disinfection.

A number of *Word* functions can also be used to lessen the possibility of contracting, or slow the spread of, viruses.

The Shift Key

The left-shift key [*Actually, either shift key will work. Ed.*] allows a file to be opened without letting any auto macros execute. This will inhibit viruses that use the AutoOpen macro to spread. Similarly, if held down at exit, the AutoClose macro will not be executed.

In order to use this feature correctly, one must be holding down either shift key at the moment *Word* is activated. Be sure to hold the key down with one hand while double-clicking with the other. It might seem obvious, but it is not an easy thing to do, even when you remember! The shift key must be held down for the duration of *Word's* startup process. Letting go early may allow a macro to execute.

Pro: Allows opening and closing documents without activating AutoOpen or AutoClose macros.

Con: Easy to forget.

Only stops auto macros when you remember to hold down the shift key.

All other macro replacements are still in place and will still infect.

Requires too much coordination and constant reminders to use effectively.

Benefits are a subset of the DisableAutoMacros method, described below.

If it fails work properly, you will not realize until it is too late.

Use Organizer to Check Documents for Macros

Macro viruses are… macros. Macros are separate from the text and are not seen unless one goes looking for them. The commonest way to check a document for macros is through Tools then Macro. Unfortunately, viruses can intercept all menu items, and a very commonly intercepted function is ToolsMacro. This makes it unwise, in a suspect situation, to select Tools, Macro to check a document for macros. In the Customized ToolsMacro section, you will learn how to create a replacement command for Tools, Macro.

It was safe, for a period, to check for macros in document files with the Organizer function. This can be reached in several ways, including: File, Templates, Organizer; Format, Style, Organizer; Tools, Templates, Organizer; or by creating your own macro to access it. Several viruses target these commands now, but it is still safe to use them if you are sure they have not been usurped.

To see if macros exist in a document file without being affected by them, exit and restart *Word* without opening any documents. If you suspect that NORMAL.DOT or the startup environment may be infected, you need to rename the file and all document and template files in the startup directory to other than DOC or DOT extensions so *Word* can be started in a pristine state. This ensures that no virus affects your viewing of a suspect file.

After starting *Word*, enter the Organizer via one of the previously mentioned methods. Choose the Macros tab. In the left side of the dialog box is a button labelled Close File. Click that button to change the label to Open File, click the button again and choose the target file in the Browse box. (By default, *Word* lists DOT files here. If the file you wish to investigate is not so named, change the Files of Type drop-down box to All Files). The filename will be shown. If any macros exist, they will be listed. If not, NORMAL.DOT will show up again with its macros. Practice with a file that you know has macros in it.

Pro: Safely determines if macros exist in the target file.

Con: You know if there is a macro, but you cannot tell if it is a virus or not.

DisableAutoMacros

DisableAutoMacros is a *Word* macro function which does what it says. If the function is invoked, no auto macros will execute automatically until it is turned off again (or until the next *Word* session). We have mentioned that many macro viruses make use of an auto function. Thus, removing the ability to execute these automatically will prevent those viruses from infecting your system.

Ironically, the best way to implement this protection is through an AutoExec macro. The following instructions produce an AutoExec macro in its own template file, not in the NORMAL.DOT file. I recommend the template file is placed in the default startup directory in order to keep the NORMAL.DOT file pristine. This makes it easier to give the file to others, and harder for viruses to find and remove.

Start by making sure your NORMAL.DOT is writable, and empty. Click on Tools, Macros. In the Macro Name box, enter AutoExec. Click the Create button then enter the following command:

```
Sub MAIN
  DisableAutoMacros 1
End Sub
```

Close the editing session. Exit and save all changes. Copy NORMAL.DOT (usually in \MSOFFICE\TEMPLATE) to the startupdirectory (\MSOFFICE\WINWORD\STARTUP) and rename it to NOAUTO.DOT. Delete NORMAL.DOT.

Pro: Disables all auto macros.

Much less chance of infection.

Con: Not foolproof.

Does not disable other intercepted macros, nor key shortcuts, etc.

Environment is no longer pristine. Others may believe the macro you have established is suspicious and this may cause technical support issues.

Prompt to Save Normal Template in NOAUTO.DOT

In creating NOAUTO.DOT in the preceding process, it is beneficial to include the command which turns on the Prompt to Save Normal Template. This is accomplished by using the following macro in place of the one above:

```
Sub MAIN
  DisableAutoMacros 1
  ToolsOptionsSave .GlobalDotPrompt = 1
End Sub
```

This insures that the option is set every time, even if NORMAL.DOT is deleted, or something resets the option. It also allows an MIS director to distribute one file which enforces two of the ideas instead of just one.

Customized ToolsMacro

As default menu items are often targeted and intercepted by macro viruses, it is important to know how to make your own menu items which will have the same functionality. These instructions create a menu item equivalent to Tools, Macro that is not usurped by a macro named ToolsMacro.

Make sure that NORMAL.DOT is writable. From the Tools menu, select Customize and choose the Menus tab. Under Categories click on Tools, then under Commands click on ListMacros. For Position on menu choose (At bottom) then click Rename. Close the editing session then exit *Word* and save all changes.

Please remember to make NORMAL.DOT read-only again. You will now have an additional entry on the Tools menu to list the macros in your document.

Pro: Bypasses the need to use Tools, Macro.

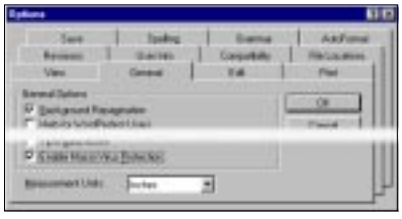
Not subject to virus payloads tied to Tools, Macro.

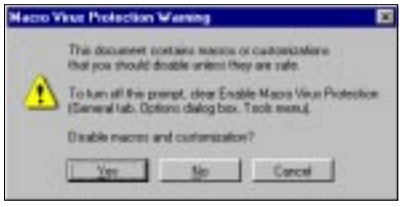
Con: Works until a virus intercepts Tools, ListMacros (but, now that you know how to do this, you can create your own toolbar choices for such things as the Organizer function).

Use Word 7.0a and Word 97

After macro viruses were discovered, *Microsoft* looked for some very basic and simple rules. It came upon this one – 'If a document does not contain macros, it cannot have macro viruses' and derived from this 'If there are macros in the document, there could be a virus'.

In both *Word 7.0a* and *Word 97* (available for *Windows 95* only), an additional configuration option was added to the Tools, Options, General dialog. If enabled, it causes *Word* to alert you should any macros exist in a document that is about to be opened. If such a document is encountered, you are given the choice of stopping, continuing as usual, or continuing with the macros disabled.





If you choose to continue with the macros disabled, *Word 97* opens the file in read-only mode and it cannot be changed without first saving it to a new name. *Word 7.0a* does not enforce this.

It is wise for a typical user to enable this option. Obviously, those who regularly use macros should probably not use it. When it generates an alert, it needs to be in relation to a virus. Many alerts which have nothing to do with viruses will 'anaesthetize' you, so you become likely to disregard the next warning, or turn the option off altogether.

Although seemingly perfect in its simplicity, *Microsoft* added some 'usability' touches to this, thus creating some security holes. These have been documented by Vesselin Bontchev and revolve around particular conditions where *Microsoft* expects macros to appear. Thus, the initial warning will not be raised (even if the macros turn out to be viral), because macros are expected to be there.

Pro: Generally effective. If there is a macro in the document, it tells you so.

Con: Can seem to alert on (almost) everything else.

People are apt to turn it off.

Read-only Recommended for NORMAL.DOT

In a following installment, methods will be presented to enforce the read-only attribute on NORMAL.DOT. However, *Word* has its own read-only enforcement for this critical file. Thus, even if the file is not read-only to the operating system, *Word* can be made to open it as if it were. The downside of this is that each time you start *Word* you are asked if you wish to open the file as read-only. This can become bothersome and lead to users turning it off. Of the techniques described in this series, this is the most irksome, as it 'interferes', even in a clean environment.

To set this option, open NORMAL.DOT (\MSOFFICE\TEMPLATES), select Tools, Options then choose the Save tab. In the lower, left corner of File-Sharing Options for Normal.dot, check the Read-Only Recommended option. Exit and save all changes.

Pro: Allows flexibility for those who would want their NORMAL.DOT to be read-only at certain times.

Con: Displays a warning message each time *Word* starts.

People are apt to turn it off.

Acknowledgments:

Vesselin Bontchev, *FRISK Software International*; Ray Glath, *RG Software Systems*; and Stefan Geisenheiner, Jivko Koltchev, Akihiko Muranaka and Francois Paget, *Network Associates*.

INSIGHT

Clued Up

Who is Graham Cluley, Senior Technology Consultant at *Dr Solomon's* in the UK and alt.comp.virus regular? 'I guess my job could best be described as Company Spokesman, although it is much more than that as I tend to get involved in all kinds of discussions and meetings internally. Some people call me "Company BigMouth" because I'm usually the person who ends up speaking to the media, and appearing on TV.'

Whatever it is he does for *Dr Solomon's Software*, Graham finds it all 'enormous fun' – 'It always surprises me that companies are prepared to pay people for doing something so enjoyable'. Above all, he likes to think that his 'audience' is learning: 'I see my job as educating people about viruses, crusading against virus hype. In a way I'm the cuddly caped crusader, fighting Good Times and PenPal Greetings everywhere I go.' Despite his background in programming, virus analysis does not appeal to him. 'I've never been put in the position of disassembling viruses – for which I'm very glad. I'm full of respect for people like Dmitry Gryaznov and Igor Muttik who do it every day. I have, however, sat in technical support taking calls from panicky users. It's very satisfying sorting out someone who thinks they've said cheerio to their data.'

Getting Noticed

Graham's sense of humour and irreverence are qualities which were appreciated by Dr Alan Solomon back in 1992. At the time, Solomon's children were playing one of Graham's computer games, Wibbling Wilf, which ended with a heartfelt plea through the PC's speaker for a job in computing, 'or at least five pounds so that I could realize my dream of leaving the supermarket with a shopping trolley full of cheesy biscuits. It also explained that I badly needed cash to visit my girlfriend in Paris and how if they didn't cough up the spondoolies it would be the end of a beautiful relationship.'

He joined *Dr Solomon's*, then just a team of thirty, in 1992 as Chief Windows Programmer. At his job interview he made the mistake of asking why there did not seem to be a *Windows* version of the *AntiVirus ToolKit*. That became his first assignment. Included in the task was the design and implementation of a *Windows*-based virus scanner and cleaner, a cryptographic checksummer, utilities to inspect files and disks at a low level, scheduler, generic cleaning tools for boot sectors and partition sector infections, and a messaging protocol between the *VirusGuard* TSR and a *Windows*-based alert screen. He also found time to help maintain the virus-finding engine of *Dr Solomon's* FindVirus for DOS and OS/2.

In September 1992, *AVTK v6.0*, the first ever *Windows* version, was successfully released. Graham remembers that not everyone appreciated it at the time, '*Virus Bulletin* poured scorn on it for having more front than Selfridges.' He continued to work in the development field, releasing the next major version of *AVTK* in mid-1993 and winning promotion to the position of Toolkit Team Leader. The integration of a Generic Detection Engine into *FindVirus* for all platforms followed later in that year.

It was at this point that Graham began to tire of the technical side of his job, and worked on developing the new skill of educating people about viruses and anti-virus software. In April 1994 he became Product Specialist for *AVTK* – 'the idea was to put someone technical in the midst of marketing and keep an eye on them.' He had spotted a significant rift in the marketing process. 'The sales people could communicate to customers, but didn't know what they were talking about. The R&D people knew what they were talking about but didn't know how to communicate it! Anyway, the solution was to make me the missing link. That way I could make sure our marketroids didn't make preposterous claims like "we can detect all past, present and future viruses". It also meant I could present the technological advances our R&D people made in a way customers could understand.'

From then on, his new position was varied and dynamic, building on a more direct relationship with the user community. His responsibilities ranged from representing the company at corporate events, communicating with corporate customers about new developments and giving corporate feedback to R&D, establishing press contacts, writing articles and technical papers, writing and checking *Toolkit* manuals and helpfiles, to giving seminars about viruses and anti-virus technology. In March 1995, he was promoted to the role of Senior Technology Consultant and began representing the company outside the UK.

During the evolution of his role into more of an educational one (or perhaps because of it), Graham's profile in the on-line worlds of Usenet News and the UK's CIX conferencing system increased.

'If you get an email from someone who can not finish a college project because of a virus, it's difficult to ignore the email or tell them to contact their current anti-virus vendor. It's much easier to try and help them there and then.'

Game for a Laugh

Graham realized early on that computers were his forte. His father had bought the family a *Sinclair ZX81* in 1980, an experience he describes as 'character building… I couldn't afford to buy software at that age so the only avenue for me was to spend hours upon hours entering code from computer magazine listings. This is how I learned to program. The *ZX81* was a great little computer: 1K of memory, no sound, no graphics (no lower case even!), and you plugged a tape recorder into the side of it to "save" your valuable programs. If you so much as sneezed it would crash and you would lose everything you had been typing.'

At school he had neglected his studies in favour of writing computer games and editing his 'Limpet Racing News', which he sold up and down the corridors between lessons. He blames these obsessions, and the discovery of the female sex, for his failure to secure any 'A' levels. Not having been offered the option of Computer Studies at school, he enrolled in a two year computing course at Guildford College of Technology, and later completed another at Bristol Polytechnic, now the University of Western England.

It was at Guildford that his computer games began to gain recognition. Having mastered programming, he concentrated on writing an adventure game called Derek the Troll (later known as Jacaranda Jim) on a *PRIME* mini-computer. It was his most ambitious to date, written in Pascal for the first time, and when Graham found a row of *IBM PC XTs* in the next classroom, he went to work converting his game to run on a PC. At about the same time, he discovered the concept of shareware and placed a message charging a small fee for hints and a map. Having arranged for some computer magazines to include it as their cover disk, he watched the registrations pour in.

The games which followed were even more successful, and not confined to shareware adventures. He also wrote a *Tetris*-clone and the *Pacman*-style arcade game which eventually landed him the job at *Dr Solomon's*. He jokes that it took him nine months to find a job because he dismayed potential employers with his intention to continue writing games, which he claimed made him more money than they were offering!

Personally Speaking

Today, Graham is more interested in countering computer viruses, and while he admits that anti-virus solutions are becoming more sophisticated all the time, it is the human element which concerns him. 'Some people say that you don't need an AV for your mail system because you have an on-access scanner that stops the virus. But that misses the true problem. The real problem is that five hundred users receive a virus via email, see a warning message and panic. That's five hundred users to calm down. If you run an email anti-virus then those five hundred people never know they were sent a virus. That can dramatically reduce the amount of time wasted dealing with viruses…'

Asked what the future holds for the anti-virus industry, Graham expressed confidence that users will always want and need some kind of computer security, no matter how the future develops. 'At *Dr Solomon's* we've recognized that you can't rely on the users to deal with viruses. You can't force your users to scan every floppy disk, every email attachment. It just won't work. So the future of anti-virus is to become more transparent, working in the background without the user necessarily realizing they're even running an AV.'

As a former games writer, Graham's attitude to virus writers themselves is perhaps more empathetic than one would expect. 'They'll get older, they'll get girlfriends, they'll stop writing viruses. Writing viruses is not a pursuit regularly followed by mature people… There's nothing wrong with writing viruses, so long as you keep them on your computer and don't put them anywhere near anyone else's. It's when you impose your virus on someone else's computer that I think it's bad. All viruses consume resources (hard disk, memory), so even harmless viruses are far from friendly. Legally? If a virus causes damage to a computer's data the owner of that data should be able to get some kind of recompense.'

In his apartment overlooking the Grand Union Canal in Berkhamsted, Hertfordshire, Graham admits to being less than organized. 'Being a single bloke means my house can get pretty untidy. This is no surprise to my work colleagues who often think that a tactical nuclear missile has hit my desk.' His private life and hobbies are as diverse as his job description. Moving to a larger house in the summer, he plans to convert an entire room into a 'book den' to satisfy his passion for reading. He has an interest in chess (he played for his school and his county) and in magic – 'not the sleight-of-hand stuff but the dramatic "I've chopped my arm off" kind of thing. We use this kind of action in our trade shows.' Despite his claims to be 'project-managing my house and fighting off international supermodels' he is very serious in his efforts to help the families of religious cult members, following his own personal experience of losing a friend to a destructive group.

Graham's may not be a household name, but he has won his place in front of the cameras as *Dr Solomon's* spokesman through sheer enthusiasm. While his schoolboy humour is infectious, his dedication to things technical is sincere, in fact it is hard to stop him once he starts: 'I am also addicted to gadgets – tiddly, tiny cameras, palmtop computers… In fact, I'm considering buying a dalek built to original BBC specifications to put in the corner of my library. I was very sorry when our events team put our radio-controlled BBC-spec K9 on the bonfire. Sorry if that sounds sad…'.

VIRUS ANALYSIS 1

Three Out of Four Ain't Bad

Jimmy Kuo of *Network Associates* has been predicting for some time that the first *PowerPoint* macro virus is approaching. He has a hypothesis that, in recent times, it takes about two years from the launch of a new, virus-supporting platform to the release of the first virus for it. There is ample evidence that loosely support this idea.

PowerPoint in *Office 97* was the first version to sport macro capabilities. Jimmy's hypothesis explains why there has not been a *PowerPoint* virus sooner. In fact, being barely eighteen months old (*Office 97* was in wide-beta quite some time before release), such a virus, appearing today, would buck the trend.

Speaking of which, anti-virus developer *Trend Micro Inc* has just announced the discovery of the first *Access* database virus. Jimmy's *PowerPoint* virus must still be under development…

Access(iV) Macros

Access97Macro/AccessiV.A, or A97M/AccessiV for short, is wholly uninteresting aside from its claim to primacy. As the name implies, it infects *Access 97* database files. The virus is not known to be in the wild, and is very unlikely ever to be so. It is written in Visual Basic for Applications (VBA), the language in which *Access 97* modules are programmed. Modules consist of procedures, functions, class definitions or some combination thereof.

AccessiV consists of one macro, perhaps not surprisingly, called Autoexec and one module, inventively named Virus to throw you off the scent. The module is a hefty twenty-three lines of Visual Basic code, although it could have been written more economically, but with identical functionality, in six fewer.

Infection Mechanism

The virus' operation is something of a departure for macro viruses. AccessiV's author has avoided some of the slightly more complex mechanisms available in *Access/VBA* that may have made the code interesting. As a result, AccessiV is a simple, direct-action infector that targets all MDB files in the current directory (more on this later) when it runs.

On opening an AccessiV-infected database, *Access* runs the Autoexec macro. This performs four functions. First, it changes the mouse cursor to an hourglass, then it changes the message in the status bar to 'AccessiV, The new Access Macro Virus, by Jerk1N of DIFFUSION'. Next, it calls the AccessiV function (in the Virus module) then resets the cursor. Mission accomplished.

The AccessiV function calls the virus' 'findfirst' and 'findnext' sub-procedures. These find all MDB files in the current directory that do not have any of the read-only, hidden or system file attributes. As each host file is located, AccessiV calls the 'infect' sub-procedure which simply copies the Autoexec macro and Virus module directly into the host file. There is very limited error checking/handling in the infect sub-procedure, but it is sufficient to prevent any problems caused by sharing conflicts and the like.

In *Access*, the 'current directory' is initially the one pointed to by the Default Database Folder setting in Tools, Options, General. However, use of the browser in certain file operations (notably File, Open and File, Save) changes the internal variable if you explicitly navigate to a directory. Simply typing a fully specified pathname into the File name field of, say, the File Open dialog, does not change this. For 'point and clickers', the current directory will be the that from which the infected MDB was loaded.

The virus makes no test for existing infections. Thus, otherwise unused MDB files in a directory containing a commonly-used, infected database will grow as new copies of the macro and module are repeatedly copied into them. In some samples the macro and module are hidden – these components retain this state when replicated.

Disinfection

Manual disinfection is simple. The 'hold down the shift key to disable auto macros while opening a file' trick works in *Access*, so open the infected database thus. Select Tools, Options, View and turn on display of hidden objects, if necessary. Delete the Autoexec macro and Virus module. Finally, reclaim any wasted space from the macro and module (especially if the file suffered multiple infections) with the Compact Database function from Tools, Database Utilities. (Note that with infected MDBs, when this function or the Repair Database function run, they close then re-open the MDB, so the virus will be run again!)

In Conclusion

A harmless, boring proof of concept, trivially demonstrating that *Access* can host macro viruses. Big deal.

A97M/AccessiV	
Alias:	JETDB_ACCESS-1.
Type:	Access 97 module infector.
Payload:	None.
Disinfection:	Remove virus macro and module from infected files as detailed in the text.

VIRUS ANALYSIS 2

No Peace on the Excel Front

Vesselin Bontchev
FRISK Software International

Late in January we received a sample of a new, interesting kind of macro virus for *Excel* from Jimmy Kuo, a fellow anti-virus researcher at *Network Associates* (formerly *McAfee Associates*). It had been found by their French subsidiary and was reported to be in the wild in France. The most interesting thing about it was that, unlike all *Excel* viruses known so far, it contains no VBA modules.

According to initial reports, it only worked under the French version of *Excel 95* and was implemented entirely as cell formulas. Its payload, invoked with a probability of 1%, changes the title of *Excel's* window from 'Microsoft Excel' to 'Enfin la paix…' ('Peace at last…' in French), so we named it ExcelFormula/Paix.A, or XF/Paix.A for short.

The Truth about Paix

Some of this preliminary information was misleading. For instance, we discovered that the virus infects and replicates happily under any language version of *Excel 95*. Further, although unable to infect a clean *Excel 97* system, if that system is already infected (for example, by upgrading an infected installation of *Excel 95*), the virus has no problems replicating under *Excel 97*, too.

Also, calling it a 'formula virus' is not quite accurate. We were worried that, if Paix was indeed implemented as cell formulas, finding it would cause a tremendous slowdown of our scanners. Given that virtually all useful *Excel* workbooks contain some kind of formula, this would mean there could not be a fast and easy way to determine that a workbook is 'clean'. Fortunately, Paix is implemented as an Excel 4.0 macro sheet. As it relies heavily on extensions to the Excel 4.0 macro language introduced in *Excel 5.0*, it cannot run under *Excel 4.0*. Excel 4.0 macro sheets have cells like a normal sheet, and its operators look very much like cell formulas, which explains the initial confusion.

Paix consists of five routines, contained in one column of a macro sheet. These are Auto_ouvrir, activation_feuille, protect, GO and Auto_fermer, respectively. The viral sheet itself is named !!!GO and marked as 'very hidden', which means you cannot 'unhide' it from *Excel's* menus. Instead, you must use either a hex editor (if familiar with OLE2 and *Excel* Book stream formats), or create and run the following short VBA module:

```
Sub Unhide()
   For Each sh in Sheets
       sh.Visible = True
    Next
End Sub
```

Running this in a Paix-infected workbook will reveal the hidden sheet !!!GO, which will seem empty. This is because the sheet is additionally protected with a password, so it cannot be modified and the formulas in it are not visible. Fortunately, breaking weak protection schemes is something of a hobby of ours, and after a little hacking, we cracked the password. To reveal the contents of the sheet, make that sheet active and select Tools/Protection/Unprotect Sheet from *Excel's* menus. When *Excel* asks for a password, use !!!GO97.

Paix's Routines

Auto_ouvrir (Auto_open in French) determines whether the system is already infected, and if not, it activates the virus. This routine disables error checking, so no error messages are displayed and execution continues should an error occur. Then it disables screen updating. Its next task is to determine whether the name of the active workbook is XLSHEET.XLA. If so, it prevents the activation_feuille routine on the !!!GO sheet from running, if that sheet resides in a workbook named TEST1.XLS. That was probably the name of the virus writer's original sample and these gymnastics would have prevented the virus from getting out of control on its author's system.

Next, Paix examines the list of all active workbooks and add-ins. If one of them is XLSHEET.XLA, it decides it has already infected the system, so it just restores screen updating and error handling. If the system is not infected, it decides which directory to install itself to. It does this by examining the first character of the full path to the current workbook. If that character (corresponding to the drive letter) is not 'C', it decides on C:\WINDOWS – otherwise it chooses the same directory as the infected workbook.

If infecting, Paix converts the current workbook (the one it is running from) to an XLSHEET.XLA add-in, residing in the directory it has just chosen. This is the operation that fails under *Excel 97* and the reason why Paix is unable to infect clean *Excel 97* systems. Finally, this routine activates the !!!GO macro sheet containing its code, uses the add-in manager to instruct *Excel* that the XLSHEET.XLA add-in is installed and restores screen updating and error handling.

The activation_feuille routine replicates the virus to new workbooks. First it obtains the name of the currently active workbook, turns off screen updating and error reporting and activates the XLSHEET.XLA add-in. Next it unhides the !!!GO sheet in both the add-in and the current workbook. If no error occurs during that last action, the current workbook contains a sheet named !!!GO, so is assumed infected. In this case, it calls the GO routine, containing its payload. Otherwise, it copies the !!!GO macro sheet from the add-in to the active workbook and saves the workbook – in other

words, it replicates. Finally, it hides the !!!GO macro sheet in both the add-in and the newly infected workbook, and restores screen updating and error handling.

The protect routine is never executed by the virus. It simply defines the variables and routine names used in the virus code, password-protects the viral macro sheet and marks it as 'very hidden'. Obviously, this routine exists because the virus writer had to perform these initialization tasks before the virus could begin spreading on its own. They could have been performed just as well by a separate macro, making the virus about twenty lines shorter and causing us to spend a few additional minutes cracking the protection password – but the virus writer obviously did not care.

The GO routine contains the payload, whose first action is to disable error messages. With a probability of 1%, it hides all the display elements normally configured via the Tools/Options/View menu, then it hides all open workbooks and changes *Excel's* window title to 'Enfin la paix…', leaving *Excel* (and the user) in a rather confused state. The last action of the GO routine is to re-enable error messages.

The last routine, Auto_fermer (Auto_close in French), is very short – just three lines. All it does is turn off error messages if the XLSHEET.XLA add-in has just been created. The purpose of this is to disable the 'Save changes in <workbook>?' question *Excel* would normally ask when closing the workbook.

Paix also contains the defined name 'noprotect'. This is hidden and attached to a shortcut (Ctrl-Z), but is not linked to any code in the virus. I suppose that such code (similar to the Unhide routine, presented above) was present during the virus' development and used to 'unprotect' the virus quickly. Paix's writer removed the code for the final release of his creation, but forgot about the name and shortcut. As a result, you can use it as a simple test of whether an open workbook is infected. Press Ctrl-Z and, if Paix is present, *Excel* will display an error message similar to this:

Language Problems

Since Paix uses French reserved routine names, how can it replicate under other language versions? The answer lies in the protect routine and in the way *Excel* treats reserved names. For instance, the protect routine defines the name Auto_ouvrir in the following way:

```
=DEFINE.NAME("Auto_ouvrir","=L2C1",2,,TRUE,14)
```

If this were executed in an English version of *Excel*, it would not have the desired effect. First, cells are referred to as =RxCy, not =LxCy and second, Auto_ouvrir has no special meaning. In Paix this line has been executed only once, by the virus author and in a French language *Excel*. In that version, the cell reference is perfectly correct and the name 'Auto_ouvrir' has a reserved meaning – it indicates an action to receive control automatically when a workbook is opened. The execution of that line instructed the French *Excel* to record in the workbook that the macro starting in the cell at the intersection of row two and column one is to be executed each time the workbook is opened.

Once this is recorded in the workbook, it is preserved even when the workbook is opened in other language versions of *Excel*, thus the macro will execute correctly. However, if Paix's source were extracted, inserted into another workbook and initialized by running the protect routine under an English version of *Excel*, it would not replicate.

Detection and Disinfection

Detection and disinfection of this new kind of macro virus poses no particular problems – even the documentation of the relevant stream formats and data structures is available from *Microsoft*. Nevertheless, it will require significant redesign of current scanning engines. Scanners should examine the Book (or Workbook in *Excel 97* spreadsheets) stream for BOUNDSHEET records describing *Excel 4.0* macro sheets. In a normal workbook there would be none of these, so it can be determined relatively quickly that a workbook is not infected. In case any Excel 4.0 macro sheet description records are found, their FORMULA records should be searched for the virus. My recommendation is to use exact identification and to checksum all BLANK, LABEL and FORMULA records of Excel 4.0 macro sheets (skipping their variable parts, of course).

Those reliant on scan strings can probably use the one in the summary box. This is rather risky with macro viruses in general and with this new class in particular. Formulae are saved as tokens that take arguments, which in Paix can vary between hosts. The supplied pattern avoids such variable arguments, but should not be used to scan speadsheets blindly. The scanner should be aware of OLE2 structures and the Book/Workbook stream formats and scan only the FORMULA records of Excel 4.0 macro sheets.

Either way, this new form of virus is not too difficult to deal with. Given its incompatibility with *Excel 97*, which *Microsoft* claims is soaring in uptake, it seems unlikely that Paix or any descendents will become a major problem.

XF/Paix

Aliases: None known.

Hex Pattern in Excel 4.0 Macro Sheets:

```
0817 065D 2121 2147 4F08 1E58
0042 01BC 001E 0100 4203 1B81
```

Payload: Hides all 'standard' displayed items and changes Excel window title.

Disinfection: Unhide 'very hidden' sheet (see text) and delete it. Once all infected workbooks are cleaned, locate and delete viral XLSHEET.XLA add-ins.

VIRUS ANALYSIS 3

Bad IDEA

Péter Ször
Data Fellows

In many cases, backups are not sufficient to recover from virus infections, as they are often already infected. If the administrator makes a mistake creating a backup batch, flawed backups could be produced for months. When recovery from backup is needed, it is often too late to recognize that the actual information on it is wrong, and if it *is* good, it is still time-consuming to restore clean environments. Thus, disinfection remains, for many, the commonest method of recovery from virus infection.

Virus writers like neither detection nor disinfection, which is why hundreds of them have already used polymorphism. Naturally, a response to this was implemented in most good anti-virus products. A few years ago, we could have spent a couple of weeks writing a special algorithm to detect a new polymorphic virus. Nowadays, such viruses only require a few minutes' work with good scanning technology. Virus writers, however, were unlikely to abandon polymorphism without a rejoinder, testing the best anti-virus products against their ideas until they found some way around them. Unfortunately, some of them also release their creations by spreading them through Internet news groups. One such virus writer is Spanska (which is Swedish for 'Spanish').

His latest creation is IDEA.6155, a polymorphic virus with three layers of encryption. The first layer is decrypted by FSE (a mutation engine). The second decryptor does not contain its own key, but is decrypted using a 'brute-force attack'. Finally, the third decryptor uses IDEA (International Data Encryption Algorithm) – one of the most secure block algorithms available to the public. The virus contains the key for decrypting its IDEA layer, so this cannot be considered a security feature. Regardless, IDEA.6155 is a resident COM, EXE, and ZIP infector, and it is difficult to disinfect. While its size is variable, it still uses stealth, even under *Windows 95's* DIR command. It is the first known virus to correctly infect 'saved' COM programs (such as CHOICE.COM) in the *Windows 95* COMMAND directory. It also includes a new form of attack against *TbScan's* validation file. Unfortunately, despite its complexity, IDEA is quite workable. During testing, I was unable to find the usual fatal bugs which prevent the spread of such viruses.

Execution of an infected program

When the virus is executed, the FSE polymorphic engine decrypts the first layer byte by byte. This involves multiple methods such as XOR, ADD, ROR, DEC, *etc*, with 8-bit variable keys. FSE executes about 100,000 instructions to decrypt the first layer. Since the length of the decryptor varies (in COM infections from 29 to 58 bytes, and from 31 to 60 bytes in EXE infections), the first byte of the encrypted layer is at a variable position. This makes detection of the virus a little more complicated.

When the first layer has been decrypted, the second decryptor takes control. This one starts testing all possible key combinations – a 'brute-force' cryptographic attack. There are four different methods to decrypt each word with four 8-bit keys – in order XOR1, ROR, SUB and finally XOR2 are tried. The maximum value of the key combinations is 16, 8, 32 and 64 respectively. The same 8-bit keys are used to decrypt the lowest and highest byte of each word. The function checks the decryption of three words at the beginning of the second encrypted layer. If the key is found, the virus uses it to decrypt the second layer, then passes control to the third decryptor. If none of the keys work, the virus terminates to DOS.

The possible key combinations for the second decryptor are restricted, otherwise decryption of an infected program could produce a noticeable execution delay. Testing the complete key space takes about five seconds on an *Intel* 386 machine, as the 'attack' needs an average of three million instructions. The code of the third, IDEA-based decryptor was written by Fauzan Mirza. The assembly source of this IDEA implementation can be found at many locations on the Internet.

In 1990, the first incarnation of the IDEA cipher (then called PES – Proposed Encryption Standard) was released. Following cryptanalysis, the authors strengthened the cipher, calling the new algorithm IPES (Improved PES). The name was changed to IDEA in 1992. IDEA's current claim to fame is that it is part of *PGP*. It uses 64-bit blocks and a 128-bit key. The same algorithm is used for both encryption and decryption. There are no bit-level permutations, but IDEA mixes three algebraic groups: XOR, addition modulo 2^{16}, and multiplication modulo $2^{16}+1$ (this operation can be seen as IDEA's S-box).

Obviously the virus cannot use brute force against its IDEA encryption layer. It contains the variable key for decryption, but the location of that key can be at two different positions in the virus, which makes full decryption more complicated for anti-virus products. Following the IDEA decryption, the virus is completely decrypted.

Installation and Payload

When an infected program runs, the virus issues its 'Are you there?' call, Int 21h AX=6969h. It assumes it is already resident if AX returns unaltered. If the system time is 15:30, the virus calls its main activation routine, which displays the message 'Warning! strong crypto inside'. Before the text appears on the screen, the virus creates the

file C:\VIRUS.COM. This is a copy of the *EICAR* Standard Anti-virus Test file normally used to test the operation of those anti-virus products that support it. In other words, it is a harmless program detected by anti-virus products *as if* it were a virus.

Since IDEA.6155 is in an endless loop at this point, the PC has to be rebooted. Many users will then check their PCs with a scanner, and this 'low-level psychology' may thus work against them. When their anti-virus product detects (and deletes) the VIRUS.COM file, they will likely assume that the problem has been detected (and fixed), while in fact, the virus is still able to spread.

If IDEA is not already active, it allocates memory for itself, marking it as allocated by DOS in its MCB. After that, it hooks Int 21h, and uses anti-heuristic tricks when calling particular interrupts to avoid detection by heuristic analysers. Then it checks the time for the activation routine, and executes the host program.

Infection

Once active, the virus monitors various DOS Int 21h functions. Interestingly, IDEA is directory-stealthed under *Windows 95* too, paying attention to long filename directory functions. The stealth mechanism is disabled when some archivers are executed (such as PKZIP, ARJ, RAR). That routine also checks the extension of the files. If it is 'ZIP', the virus adds an infected README.COM into the archive. Three small demonstration programs are used as hosts. These were probably not written by IDEA's author and contain advertisements for WWW sites and BBS systems.

The virus infects COM and EXE files during execution, but avoids infecting COMMAND.COM and several well-known anti-virus programs. Its self-recognition mark is the value 69h ('i') at offset 03h in COM files or at offset 12h in EXE files. It checks whether the potential host is an EXE and infects appropriately. It does not infect COM programs shorter than 1000 bytes or longer than 57,000 bytes.

IDEA pays particular attention to 'saved' *Windows 95* programs that have 'ENUNS' protection. There are a few viruses (such as Memorial, see *VB*, September 1997, p.6 and December 1997, p.9) which avoid infecting these programs, but IDEA is the first known virus to infect them without any problem. It simply moves the 'ENUNS' string to the end of the infected file and patches the value of the word after it. It is much easier than it sounds. This protection is bypassed by adding the difference in size between the infected and original file to the original check value. So much for *Microsoft's* protection scheme.

The minimum virus size is 6155 bytes. This includes the shortest possible FSE decryptor, the actual virus, and a random value calculated from the original timestamp (up to 29 bytes). The polymorphic engine is called to create the first decryptor. The other encryption layers are then applied and finally the whole bundle is written to the host.

If infecting a COM file, the first four bytes, including the self-recognition mark, are modified in order to point to the new entry point. With EXE files, the infection marker is at offset 11h. IDEA modifies the CS:IP fields of the EXE header to point to its own code. IDEA handles file attributes correctly, infecting read only files properly. It also restores the program's original date-stamp.

Modification of ANTI-VIR.DAT

Several viruses avoid detection by integrity checkers by deleting their checksum files. However, IDEA avoids quick recognition by patching this file instead. This attack is targeted against *ThunderByte's* ANTI-VIR.DAT checksum files. As the names of checksummed files are not encrypted in this file, the virus easily patches the first character of its host's entry after infecting it. The integrity checker will not alarm on the file change, treating the (now infected) file as not having been checksummed. The next time the checksumming procedure runs it will create a new checksum entry, but for the infected program. This shows that it is generally better to encrypt such checksum files, although this may reduce the speed of the product somewhat.

Conclusion

On the DOS front, we can expect to see the appearance of more, and more complex, polymorphic viruses which are hard to detect and harder to disinfect. The bad news is that the day of the complex, workable polymorph seems to have arrived. This represents new challenges for anti-virus vendors in 1998, particularly those that implement thorough disinfection.

IDEA.6155

Alias:	Spanska.6155.
Type:	Resident, stealth, polymorphic.
Infection:	COM and EXE files. Adds an infected README.COM to ZIP files.
Self-recognition in Files:	69h ('i') at offset 03h in COM files and offset 12h in EXE files.
Hex Pattern in Files:	Not possible.
Hex Pattern in Memory:	`5055 8BEC C746 0200 405D 58CD 21C3 B43C 33C9 CD21 C3B4`
Payload:	Displays an animation at 15:30 each day an infected file is run. Causes *TbScan* to revalidate infected files.
Removal:	Under clean system conditions, restore infected files from backups or replace with originals.

PRODUCT REVIEW

eSafe Protect for Windows 95

eSafe Technologies claim that *eSafe Protect* is something of a panacea for the ills to be found lurking outside our computers, which all anti-virus vendors would have us believe are constantly seconds away from disaster. Whether the world is such an unpleasant place as all that is debatable, but too much protection is indubitably better than too little. Allegedly, *eSafe Protect* guards against not only viruses, but also hostile Java and ActiveX components, Trojans in their many guises, and attacks by hackers. It can even act as a content controller while surfing the net. Detection of viruses and Trojans covers email and FTP as well as the more standard areas checked by other programs. How close to reality are all these claims? Read on.

Thirty-day evaluation copies of *eSafe Protect* are currently in wide circulation in the UK as part of a strong advertising push by *eSafe*. Those only differ from the reviewed version in their lack of registration codes and cards. There is also a corporate version on the market, which includes additional management tools and the like.

It is necessary for *VB* to make a disclaimer at this point, just like the inevitable legal documents hidden in software boxes. Although the grand vista of *eSafe Protect's* features was addressed fairly thoroughly, this review is only intended to be authoritative on the way in which the anti-viral parts of the package are implemented and integrated into the greater whole. The testing protocols used for some of the features claimed were necessarily of an *ad hoc* nature, and neither sufficient time nor network resources were available to test every possible combination of events.

eSafe's main window, with its Protection and Threat meters, and Protection Shifter. An option to display just the meters saves desktop real-estate while allowing continuous monitoring.

Packaging and Documentation

eSafe Protect's packaging is a vision in blue and orange, with a surfing motif, possibly designed to shock the more aesthetic of users into submission. The box the reviewed software was in contained a ninety-page manual, a multi-part registration card and a jewel-cased CD in a sealed envelope. The licence agreement on this envelope is of the kind that claims your compliance by your opening of the envelope, though mention of truly legal matters is reserved for the manual and installation routine.

The registration card offers the option to request information concerning the corporate version of the software, *eSafe Protect Professional*, of which more later. More urgently, it is necessary to register, as the manual states explicitly that unless the software has been registered you will not be entitled to technical support.

The card is also the location of the installation code (needed for a fully operational product), all but hidden on the quick start guide which summarizes the installation procedure. The manual is written in a friendly, conversational style, with some idiosyncratic turns of phrase. The layout is logical and easily followed.

Installation

Primarily a *Windows 95*-oriented product, it should not be surprising that the *eSafe Protect* CD autoruns the installation procedure. The first option you are presented with is a choice of languages – English, Swiss German, Standard German, French and Hebrew. This is followed immediately by the standard legal agreement.

The hyperbole concerning mighty protective abilities is, as with so many other products, shattered at this point, with such phrases as 'the Product, if operated as directed, will substantially perform in accordance with the functionality described in the Documentation' juxtaposed with '*EliaShim* does not warrant, however, that your use of the product will be uninterrupted or that the operation of the Product will be uninterrupted or secure.' For a product including a personal firewall, this pair of statements would seem to constitute a direct contradiction.

Installation proceeds with the option to choose a directory in which to store the software. Oddly, the program recommends short pathnames (8.3 and no fancy characters or spaces), and the installation process refuses to continue if a long pathename is suggested. It is possible to register on-line during this process, or to upgrade from the evaluation version to the full version, paying via credit card. The *eSafe* website is replete with *possible* terrors available on the net, just in case the user is not sure that *eSafe Protect* is a necessity of modern life.

A further option is dependent upon whether a scanner other than *eSafe Protect's* will be the source of on-access scanning, in which case only the on-demand module need be installed. Of more dubious usefulness is the standard option not to install on-access scanning within the DOS box. Scanning in DOS is considered an 'advanced option', and must be selected specifically. Since most non-macro/non-Boot viruses are only likely to be a danger in a DOS box, this seems a strange choice indeed. *eSafe* explains that this allows integration of *eSafe Protect* with on-access scanners from other vendors.

During the installation process you are also given options to run a complete scan of your PC for viral activity, and to create a rescue disk in case disaster should strike later. Every PC should have an emergency boot diskette, and many scanners these days provide assistance in producing them. For users accustomed to *Windows* GUIs, recovery diskettes are not the most friendly of creatures, but this observation is common to most rather than a criticism of *eSafe's*. Some explanation of the contents, and how to use them in anger [*or despair? Ed.*] would be of benefit, though non-experts faced with a diskette loaded with such things as REGEDIT, FORMAT and SCANDISK are presumably likely to call Technical Support! The disk appears to contain information concerning system configuration, but from the manner in which it is stored, it is not at all apparent exactly what this information is. As with the sandbox, this may prove only to be helpful to the patient user with a good deal of background knowledge.

The Front End and Sandbox Theory

The program having been installed, presentation is the first noticeable feature. The windows used for information and configuration are pleasantly laid out, with the on-line threat-meter a fine addition to any desktop. We imagine these could inspire competitions of the 'who can obtain the highest danger rating' type around the office, but they provide an immediate indication as to the developer's impression of the riskiness of your computer use.

Upon *Windows 95* initialization, *eSafe Protect* scans for possible security problems in a way which is central to the philosophy of the product. The anti-vandal component attempts to detect newly installed applications and plug-ins which could pose a security threat. A number of more common products, *Netscape Communicator*, *Microsoft Internet Explorer* (*MSIE*), *Microsoft Netmeeting* and the like, are detected automatically, and the patented 'sandbox' technique applied to them. The sandbox imposes a 'security profile' on an application. It does this by monitoring file, and possibly network, I/O and enforcing a set of restrictions upon the application, depending on its stated role.

For example, *MSIE* is allowed full access to the TEMP and RECYCLED sub-directories, and has access to the *Windows* directory and its sub-directories (apart from FONTS, HELP and SYSBACKUP) but is not able to delete anything in these. The COMMAND sub-directory is also an allowed

Configuring a Resource Protection Set in *eSafe's* Advanced Configuration mode is a case of defining what sort of file access is allowed in which directories on your PC.

area, yet here no writes or deletes are permitted. Any other sub-directories are considered outside the normal operation of an untampered version of *MSIE*, and therefore no access of any sort is allowed in them. If access is attempted, there are options simply to deny access, or to offer the user a choice as to whether the activity should be allowed this once, always or never.

This is the primary method by which *eSafe Protect* guards against what *eSafe* terms 'vandal' attacks. Any ActiveX applet launched by *MSIE* should be constrained by these limitations so as to be unable to access, alter or otherwise behave wickedly in an unauthorized folder. Newly created directories are automatically given the highest level of protection. For a new program, which might well be expected to need access to further unpredictable areas or files, there is an option for a learning period for a new application. During this time, additions may be made to the legally accessed areas for this application.

One niggle is in the treatment of drives other than the primary hard drive. Due to the nature of *eSafe Protect's* implementation of the sandbox, it can only be applied to removable disks at the folder structure level, and thus floppy drives and removable large media drives (such as Zips and LS-120s) are unable to be protected on an individual basis. More importantly, the list of known, 'worrisome' applications does not seem very comprehensive. For example, *Turnpike*, supplied with *eSafe* in a popular CD bundle in the UK, the recently headlined *mIRC* (see our feature article on p.7), and the popular *Pegasus Mail* email program were all unknown to *eSafe Protect*.

To protect an unknown application, the user is required to set up the basic sandbox parameters, thus being required to second guess the areas legitimately used by their newly installed software. There is some latitude for improvement in the choice of default sandbox settings too. For example,

Pegasus Mail was initially denied access to reading *eSafe Protect's* virus definition files, thus preventing on-access scanning of documents received as email attachments.

The learning process can be set to either silent or interactive mode, according to the user's wishes. Under the former setting, one week long by default, all file operations by the application are considered valid and extend the sandbox for that application. Taking the second option of bypassing the inbuilt learning process, allows you to be as paranoid or liberal as you see fit. This option assumes a greater knowledge of what is likely a valid file operation from a given application. It is also a good way to 'see' what a program is up to – eavesdropping on the file I/O activity of even some 'simple' applications can be quite an eye-opener.

Both choices are problematic. Only a completely trusted program should be allowed free rein for a week, which means the sandbox is almost irrelevant, providing little or no protection should your trust be misplaced. On the other hand, deciding whether particular files or folders may be accessed on a case by case basis quickly becomes irksome, even for simple processes. Testing and configuring *eSafe Protect* in a corporate environment might be very onerous indeed. One step to alleviating possible problems here would be the provision of a much larger set of pre-defined application profiles.

That said, the system did, by and large, work. A hostile ActiveX component downloaded in a page on a test web server was spotted easily and the applet thwarted in its attempts to read, move and finally delete files from the local hard drive. Similarly, more direct attempts to delete files from within *MSIE* were logged and prevented.

The results of our tests of *eSafe Protect's* abilities against some Trojan Horse programs were less clear-cut. This was due in most cases to the Trojans tested being AOL-specific and it not being possible to set up the entire AOL system on the *Virus Bulletin* test net. It should be noted, however, that known Trojans are not detected *per se*, as is the case with those scanners which detect Trojans. With *eSafe Protect* active, Trojans are brought to the user's attention through their attempts to access areas outside the parent application's sandbox. Thus, a Trojan from within *MSIE*, for example, and later running it from the desktop or a DOS box affords no protection from the Trojan's effects unless you explicitly sandbox it. However, attempting to launch a Trojan from *MSIE* should afford a degree of protection.

With the push to integrate views of the web with those of the local machine and network, settings that prevent malicious applets causing trouble may have bothersome side-effects.

This testing, however, turned up an issue of some concern. Some of the tests were run on a *Windows 95* machine with *Internet Explorer 4.0* (*IE4*) installed and the active desktop option enabled. *eSafe Protect* treated *IE4* to the default sandbox for *Internet Explorer*, thus preventing much of the normal use of the active desktop! *IE4's* 'integration' of the web with your local machine may be problematic for the *eSafe Protect* approach. It has no way of knowing that *IE4's* wish to stroll through your *Windows* system directory is the result of a normal user action, and thus allowable, or that of an Internet-borne nasty, and thus to be intercepted.

Running eSafe Protect

As described earlier, when *eSafe Protect* runs as part of your *Windows 95* startup, it checks for new threats and updates its configuration. The main application window provides control of overall system protection, offering zero, minimal, standard or extreme levels, and access to the configuration on-demand virus scanning and fine-tuning of the other protective measures. Only this last GUI triggers the requirement for the optional password to be input, though access to the configuration may be varied, for example allowing only an adminsistrator the right to change configuration, and also allowing differing Web access and content screening options to be applied on a per user basis.

Content filtering was not tested. It relies on detection of text strings and words in URLs, newsgroup names and in the body of email messages, and the developers claim it is not a major strength of the package. The, perhaps obvious, example given in the manual is 'sex'. Although this might prevent access to get-sex-here.com, it would possibly anger innocent users of sextant-communication.com. Also not tested were the port-blocking features, whereby you can prevent a program accessing any TCP/IP ports.

eSafe Protect's Scanners

We are much more on home ground when testing the virus scanning components of the product. These proved to be consistent in operation in their on-access and on-demand modes. Having set up the test network for the more informal web testing we also downloaded the *VB* test-sets by FTP and the other aspects of *eSafe Protect* did not interfere with the on-demand scanner's interception of this.

On *VB's* virus detection tests, *eSafe Protect* shows a slight improvement over the recent results of *ViruScan* (which incorporates the same *EliaShim* scanning engine). Of particular note was the detection of every macro virus in the

test-sets. Even more happily, past false alarm problems seem to have been laid to rest – the clean test results showing no false positives. More surprising results arose when the various options available for on-demand scanning were scrutinized. The on-demand scanner has two scanning modes – full and smart. Though it might be expected that the smart mode would be faster than the full and the latter might find more viruses, this was a rather naive assumption.

The standard configuration for *eSafe Protect* takes the common approach of checking those files most likely to contain viruses – in this case only files with extensions of COM, EXE, XL? and DO?. This set may be edited to suit your needs. Full scans, however, look at all files. As the *VB* Clean test-set (used for timing and overhead tests) only contains files with these extensions, any time difference between smart and full mode should reflect nothing but differences in scanning techniques. Since *eSafe Protect* creates checksums for scanned files, at least two timing runs were required, with a third used as a consistency test.

Results

The results show that the full test mode has a consistent 3.0 MB/s data rate on our test machines, with no change in speed being noted on the second or subsequent runs. The first smart check, where checksums were being produced, clocked in at 2.0 MB/s, though subsequent tests were consistent in scanning some 3.6 MB/s. An overhead test similar to those in recent comparative reviews showed the on-access scanner's overhead, with default settings, to be in the region of ten to fifteen percent.

This shows a definite, though perhaps less than expected, improvement in speed due to the use of checksumming. *eSafe Protect's* checksum files are stored one per directory, rather than in a central repository. They are not hidden, but are flagged read-only. The default setting of (re-)creating a checksum file automatically and with no notification is a security risk, allowing a virus (or 'vandal') to simply delete *eSafe Protect's* fixed-name checksum files after infecting or otherwise wreaking havoc.

In the virus detection tests, *eSafe* missed two of the polymorphic Neuroquila.A samples in the In the Wild File set, giving 99.7%. This possibly reflects the 90.9% result against the Polymorphic test-set – *eSafe Protects's* weakest performance in these tests. Missing two of the Hare variants in the In the Wild Boot test-set resulted in a 97.8% detection rate. Thus, the combined In the Wild result was 99.4%. The two Cruncher samples were all that prevented perfect detection of the Standard test-set resulting in a 99.8% score. *eSafe Protect's* perfect detection of the Macro test-set leaves little room for futher comment there.

Detection of macros in emailed documents was also tested, with the results here in line with the figures above, although time constraints made it possible to test a subset. Multiple attachments were handled successfully, and the disinfection routine, not surprisingly, only targeted the detatched files.

Conclusion

eSafe Protect combines an improving anti-virus product with other system protection tools in an inexpensive and well-integrated product. However, it is not without flaws. The sandbox technique is akin to generic virus detectors which only apply heuristics in their search.

As a method of virus and other malware control, this is not entirely satisfactory as the user has to be knowledgeably involved in the detection process. This undermines the value of the product to users expecting automatic protection mechanisms. This is not to say that *eSafe Protect* is of no value; we certainly learned quite a bit about the inner workings of some of our software during testing. We have, in the past, seen behaviour blockers and active monitors largely fail to gain widespread acceptance. This is usually due to their constant alerting and the user involvement required in the use of these products.

While the violation messages produced by *eSafe Protect* are useful, they are of maximum benefit to those who know what they mean. To the less computer literate they will be downright confusing. That administrator rights may be used to limit these messages and simply deny access helps little if less knowledgable users become frustrated by their inability to perform simple actions for unknown reasons. The temptation to turn it off could mount quickly.

It is not the fault of *eSafe Protect* that this is the case, but the nature of the product is such that a good deal of knowledge is required to make the most of it, and a fair amount of time must be spent to ensure that new applications are rendered safer whilst remaining usable. Ultimately, *eSafe Protect* does do what it claims, but only in the hands of a user who has an understanding of the dangers against which it is applied and the willingness to persist with its alerts.

Technical Details

Product: *eSafe Protect v1.1 for Windows 95.*

Developer: *EliaShim Ltd*, 22 HaAshlag Street, Haifa 35022, Israel, Tel +972 4 8728899, fax +972 4 8729966, email sales@eliashim.co.il, WWW http://www.eliashim.com/.

UK Vendor: *eSafe Technologies (UK) Ltd*, Premier House, 112 Station Road, Edgware, Middlesex, HA8 7BJ, England, Tel +44 181 3811923, fax +44 181 3811924, email sales@esafe.co.uk, WWW http://www.esafe.com/.

Availability: This program requires 8 MB of memory and 5 MB of free disk space.

Version Evaluated: Version 1.1 Standard.

Price: Standard version, licence price for single user $69.99 or £49.95. Enterprise version, licence price for 25 users £995. UK prices are inclusive of VAT.

Hardware Used: Two 166 MHz Pentium-MMX workstations with 64 MB RAM, 4 GB hard disk, CD-ROM drive and 3.5-inch floppy drive running *Windows 95 (SP1)*. The workstations could be rebuilt from sector level disk images. These were linked to a *Windows NT4.0 Server (SP3)*, running WWW, POP3 and SMTP mail services.

Test-sets: Complete listings of the test-sets used are at http://www.virusbtn.com/Comparatives/NT/199803/test_sets.html.

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, The Pentagon, Abingdon Science Park, Abingdon, Oxfordshire, OX14 3YP, England

Tel 01235 555139, International Tel +44 1235 555139
Fax 01235 531889, International Fax +44 1235 531889
Email: editorial@virusbtn.com
World Wide Web: http://www.virusbtn.com/

US subscriptions only:

Virus Bulletin, 18 Commerce Way, Woburn, MA 01801, USA

Tel (781) 9377768, Fax (781) 9320251

END NOTES AND NEWS

***Norman Data Defense Systems* opened its new UK headquarters in Milton Keynes, Buckinghamshire on 28 February 1998.** With more than two hundred employees worldwide, *Norman* is now represented in the USA, Europe, Asia and Australia. The company's main offices remain in Lysaker, near Oslo, Norway. For more information, contact the UK branch; Tel +44 1908 847410, fax +44 1908 847552, or email info@normanuk.com.

Following its takeover of *Pretty Good Privacy* in December 1997, ***Network Associates* is to acquire *Trusted Informations Systems* *(TIS)*** in a stock for stock merger valued at over $300 million. With this deal, *Network Associates* is set to become the largest security software company in the industry. More details can be found at the organization's Web site http://www.nai.com/.

The Chinese Government is putting the nation's police force in charge of all computer virus research. According to the Xinhua News Agency, organizations wanting to study viruses or develop anti-virus software must register with the Ministry of Public Security (MPS). Individuals and companies with computer-related criminal records will be prohibited from collecting or storing viruses for two years. US-based researchers at *IBM's Watson Research Center* are sceptical: 'Trying to prevent a new virus from [infecting] a network would be like trying to prevent the 'flu.'

Integralis* has announced the imminent release of *MIMEsweeper for Lotus Domino. This product reflects further development in the partnership between *Integralis* and *GROUP Gmb.H*, combining *MAILsweeper* and *WEBsweeper* content security components with Notes-specific *WatchDog*, *BodyGuard* and *Chinese Wall* modules. For details, email info@mimesweeper.com or contact Catherine Jamieson; Tel +44 118 9306060.

Further to its acquisition of *Virex* from *Datawatch Corporation* in October 1997, and successful sales in the US market, ***Dr Solomon's* announce the availability of *Virex for Macintosh* in the UK.** The product's Scan-At-Download technology automatically protects users from Internet and email-borne viruses, while the *Virex Administrator*, available with multi-user licence packs, scans and updates across a network. *Virex for Mactinosh* is available from *Softline Distribution* at £99 excluding VAT for a single user pack; Tel +44 181 4011234.

The ICSA's conference, *IVPC '98: Protecting the Workplace of the Future*, will take place at Lake Buena Vista, Florida from 28–29 April 1998. For the first time it will run concurrently with a new event. *Remote Access: Building and Managing the Workplace of the Future* is to be presented by *GartnerGroup*. For more information about registration discounts and availability contact Ashley Pearce; Tel +1 203 316 6757, or email ashley.pearce@gartner.com.

***Sophos* has developed *Sophos Anti-Virus Interface* (*SAVI*), as part of its anti-virus range.** Running under *Windows NT*, *SAVI's* DLL architecture dispenses with database initialization and memory requirement problems, using a single copy of the virus database to process all requests. Allegedly, *SAVI* increases performance over command-line scanners up to thirty-fold. *Integralis' MIMEsweeper* is the first *SAVI*-compliant product. That company's Technical Director, Andy Harris, called *SAVI* 'one of the most important developments to come out of the anti-virus vendors for some time'.

***F-Secure Anti-Virus Macro Control* from *Data Fellows* works on a certification principle**. All unknown macros that have not been certified by the administrator, including macro viruses, will be locked out. The product checks files for third party and commercial templates in addition to generally known macros. Contact Mikko Hypponen; Tel +358 9 859900, fax +358 9 85990599, for details.

An introductory computer virus workshop on 13 May 1998 will be followed on 14 May by an advanced session at the *Sophos* training suite in Abingdon, UK. To register for a place on the course, contact Karen Richardson; Tel +44 1235 544015, fax +44 1235 559935, or find details at http://www.sophos.com/.

Network Security Management* is now trading as *Network International, with offices in London, Edinburgh, Berlin, Oslo, Stockholm, New York, Jamaica, Antigua and Sydney. Edward Wilding, Director of Computer Evidence and Investigation and a Consulting Editor of *Virus Bulletin*, explains that the new name better reflects 'the substantial growth of the company's business, both in terms of our range and geographical coverage'. The organization's services now encompass fraud risk management, pre-employment screening, electronic countermeasures, data interrogation, forensic sciences, computer evidence and revenue enhancement.