AUGUST 2003

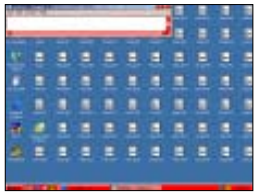
virus BULLETIN

The International Publication on Computer Virus Prevention, Recognition and Removal

CONTENTS

ISSN 0956-9979

IN THIS ISSUE

SPOT THE DIFFERENCE

Hapless users double-clicking on what they believe to be a screensaver of adult movie star Maya Gold, will be confounded when they find a desktop full of zero-length files instead. Gabor Szappanos describes this highly localised Hungarian virus and concludes that virus writers skipping their English lessons is something for which we can be thankful.

page 4

RETURN OF THE MACRO VIRUS

The days when macro viruses dominated the virus prevalence charts may be long gone, but June 2003 saw the appearance of W97M/Lexar.A, a macro virus that uses a simple method to bypass *Word*'s macro virus protection. Richard Marko provides the details.

page 10

COMPARATIVE REVIEW: NETWARE

Matt Ham finds that issues which made life difficult during last year's *NetWare* comparative review have simply evaporated, to be replaced by features which are actually useful. He tests the abilities of 11 products for *NetWare*.

page 17

'Microsoft *would be extremely foolish to bundle a virus scanner with* Windows.'

Juha Saarinen
Independent technology writer

THE SUBTLE RETURN OF MSAV

Microsoft has a long tradition of releasing initially very mediocre software (I'm being polite). Remember the first 16- and 32-bit versions of *Windows*? DOS 4.0? Early *Internet Explorer*? These are just a few examples of *MS*-ware from which the world should have been spared. However, the Redmondians don't give up easily. Through selected technology purchases and plenty of polishing work, they eventually develop what started off as lame software into products that are good enough for users to see no reason to swap to competitors' wares.

Microsoft's venture into the anti-virus field is a glaring exception though. The *MSAV* scanner developed by *Central Point Software* and bundled with DOS 6.x was the sort of resounding fiasco that you would imagine *Microsoft* would like to forget rather than repeat. Instead, *Microsoft* has gone and bought *GeCAD*, in response to customers who 'told us they needed a safer, more trustworthy computing experience'.

Predictably, the *GeCAD* acquisition has been slated by industry commentators. Former *VB* editor Nick FitzGerald calls the *GeCAD* purchase 'a fundamental mistake on *Microsoft*'s part.' He says, 'It shows no clue of how modern, scanning-based AV technology works and what the basic weaknesses are.' Nick adds that, if *Microsoft* goes ahead and incorporates *RAV* into *Windows*, it will become the anti-virus most targeted and attacked by VXers. By deploying *RAV* into the most virus-prone market of them all – home user and SoHo – *Microsoft* could end up suffering bags of bad PR.

Given the above, *Microsoft* would be extremely foolish to bundle a virus scanner with *Windows*. Apart from being unworkable and detrimental to users, it would be an invitation for various monopoly watchdogs to sink their teeth into *Microsoft* once again.

In fact, although statements from *Microsoft*'s Security Business Unit about future anti-virus and security directions are vague, it looks like the *GeCAD* purchase will not result in *MSAV Mk II*. But, before you heave a sigh of relief, read up on 'Windows File System Filter Manager Architecture' (let's call it WFSFMA), because that's where *GeCAD*'s expertise will be deployed. In essence, *Microsoft* will provide a 'core engine' for which anti-virus developers then write file system filter drivers.

Buggy third-party drivers interacting with the operating system at a low level have long been a headache for *Windows* users, so there is some justification for *Microsoft* to create a standard set of APIs that govern how anti-virus programs should interact with the file system. However, dealing with buggy drivers by preventing direct access to operating system internals is one thing. Handling deliberately malicious code in the same fashion is another matter altogether.

Even with *GeCAD*'s help, does *Microsoft* really have the experience and in-depth knowledge required to create an effective architecture to protect against malware? Look at the bundled firewall in *Windows XP* – it didn't occur to *Microsoft* to give it IPv6 support. What's to say that the mini file system filter drivers won't be hobbled by similar architectural omissions? Meanwhile, other commentators have pointed out that *RAV* is a good solution for *Linux* and UNIX plus clones, but not so popular with *Windows* users. Yet *Microsoft* bought *GeCAD* to develop anti-virus defences for *Windows*.

If *Microsoft* decides to make compliance with the new APIs in WFSFMA a condition for *Windows* Logo approval, they will likely serve to thin the field of anti-virus developers. The vendors of other third-party system utilities ended up with precious little ability to distinguish their products from those of others after API changes in *Windows* were implemented.

Of course, I could just be flaunting my deep ignorance of what WFSFMA is all about. However, unless *Microsoft* somehow garners divine foresight of malware writers' intentions, this looks like a step in the wrong direction. *Microsoft* would be wiser to put money into mending the broken security model in *Windows* that necessitates anti-virus solutions in the first place.

NEWS

NO MORE MR NICE GUY: UK GETS TOUGH ON HI-TECH CRIMINALS

London's Court of Appeal has turned down Welsh virus writer Simon Vallor's appeal to shorten his two-year custodial sentence. Vallor, who was convicted on three counts of releasing a computer virus contravening the Computer Misuse Act 1990, claimed at the time (as is the virus writer's wont) that he had been ignorant of the extent of damage his actions would cause. At the Court of Appeal, Vallor's counsel argued that the 22-year-old's sentence should be shortened on account of his relative youth, previous good character and the fact he cooperated with the police.

Unfortunately for Vallor, the judges failed to be moved by the argument. Mr Justice Aikens deemed Vallor's acts to be both calculated and disruptive and dismissed the appeal. *VB* applauds the Court of Appeal for standing firm – the deterrent effect of the few cases in which virus writers have been successfully prosecuted appears to be low already, without offenders being let off their full sentences for such insubstantial reasons.

Meanwhile, the British Home Office has announced plans for a new e-crime strategy aimed at increasing the number of hi-tech criminals that are caught and prosecuted. The strategy will focus on both old crimes committed using new technology, and newer crimes, such as denial of service attacks and hacking. It will also provide an analysis of the current and likely future nature of e-crime, producing a framework for Government, law enforcement agencies and industry, and will ensure that existing international agreements, such as the EU Framework Decision on Attacks Against Information Systems stand up to the challenges posed by hi-tech crime. Junior Home Office Minister Caroline Flint said: 'The Government has invested £25 million in combating hi-tech crime, setting up the National Hi-tech Crime Unit, within the National Crime Squad, and helping local forces to fight e-crime. But we need to do more, and [to] coordinate and focus our efforts. Our e-crime strategy will bring together industry and law enforcement agencies to deliver an enhanced and robust response to the prevention, detection and prosecution of e-crime.' The strategy is scheduled to be produced by February 2004.

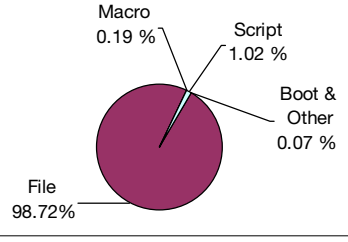
With recent reports (see http://www.sophos.com/pressoffice/pressrel/uk/20030630topten.html) suggesting that the number of new viruses in the first six months of this year is up 17.5% on the same period last year, any plans to crack down further on virus writers are certainly welcome. However, whether virus writers are likely to be deterred by this news would seem doubtful – we can only hope that once the strategy is put into effect we will see a significant increase in the number of successful prosecutions and a corresponding impact on the virus-writing community.

Prevalence Table – May 2003

Virus	Type	Incidents	Reports
Win32/Sobig	File	10,874	60.48%
Win32/Klez	File	2,867	15.95%
Win32/Bugbear	File	2,745	15.27%
Win32/Fizzer	File	300	1.67%
Win32/Yaha	File	245	1.36%
Win32/Gibe	File	160	0.89%
Redlof	Script	94	0.52%
Win32/Ganda	File	76	0.42%
Fortnight	Script	66	0.37%
Win32/Opaserv	File	62	0.34%
Win32/Lovgate	File	60	0.33%
Win32/Magistr	File	59	0.33%
Win32/Holar	File	34	0.19%
Win32/Nimda	File	27	0.15%
Win32/Valla	File	27	0.15%
Win32/Funlove	File	24	0.13%
Win32/Hybris	File	17	0.09%
Laroux	Macro	16	0.09%
Win32/SirCam	File	16	0.09%
Win32/Deborm	File	15	0.08%
Win32/Parite	File	14	0.08%
Win32/Elkern	File	11	0.06%
Win32/BadTrans	File	10	0.06%
Win32/Dupator	File	10	0.06%
Win95/CIH	File	8	0.04%
Others[1]		142	0.79%
Total		17,979	100%

[1]The Prevalence Table includes a total of 142 reports across 53 further viruses. Readers are reminded that a complete listing is posted at http://www.virusbtn.com/Prevalence/.

Distribution of virus types in reports



VIRUS ANALYSIS 1

NOT ALL THAT GLITTERS IS GOLD

Gabor Szappanos
VirusBuster, Hungary

It has been quite some time since the last Hungarian virus made it onto the WildList (the last one I can remember was WM/Mentes, back in 1998). It was out of this silence that the Magold virus variants came. Magold did not top the international virus charts, but caused massive infections in Hungary.

Magold is one of the very few highly localized viruses – despite being a mass-mailer with a very aggressive address-collecting routine, samples of Magold were captured in Hungary only.

There are five known variants of the virus. The first three are very similar, with only minor modifications to the original version, while several signs indicate that variants D and E were written after Magold.A hit the news.

All variants were written in Borland C++ Builder and compressed with unscrambled UPX (the compressed length of the five variants are, respectively: 240,640 bytes; 241,152 bytes; 240,640 bytes; 239,104 bytes and 238,592 bytes), the uncompressed versions were not observed in the wild. None of the variants work under *Windows 9x* systems, as they rely on the existence of PSAPI.DLL.

The main method of propagation of this virus is via email messages, but the virus can spread via IRC and peer-to-peer file exchange as well. The virus also copies itself to the floppy disk as Maya Gold.SCR.

INFILTRATING THE SYSTEM

Magold uses a mutex named 'AZERT SEM KOSZONOK BE BE BE! SOT! EBBEN SINCS KOSZONET! ---' to make sure that only one copy of the worm is running.

Versions D and E use the mutex 'EZ A MAGOLD NEV TETSZIK! DE MI AZ AZ AURIC? ---raVe--4-' (a rough translation of which is: *'I like the name Magold, but what is Auric?'* – a very good question indeed), proving (by the inclusion of the virus name Magold) that these variants were written after Magold.A had been detected by the virus scanners.

During execution the virus displays a camouflage error message.

For the A, B and C variants the message is:

```
DirectX error!
Address:0002R1A9V8E52000
```

For the D variant the message reads:

```
ANT_MAGOLD_V1
A telepítés befejezõdött!
```

(which translates as: 'installation complete')

And, for the E variant the message reads:

```
DirectX Error!
Address:19851022
```

It is highly likely that the last message shows the virus author's date of birth (19851022) – note that the year appears in the first message as well, as 1985 is mixed with the letters of RAVE.

The virus connects to the ftp server ftp.fw.hu with a hard-coded user name ('theoffspring' for versions A, B and C, and 'dread_punk' for D and E) and password. It downloads the file verz.txt, and the commands contained in this file direct the virus.

The possible commands are: email, irc and halozat. The first two are obvious, the third is for infecting mapped network drives. The 'day of the week' value of the completion of these commands is stored under the registry hive \HKLM\RAVE (A,B,C), or \HKLM\DREAD, along with the infection date. At the time of writing the accounts are disabled (at least they are not accessible with the burned-in username-password combinations).

Version E also tries to download and execute a file (update.exe or update.scr) from ftp.fw.hu.

Next, the virus copies itself into the Windows directory as RAVE.EXE (DREAD.EXE in versions D and E) and 'Maya Gold.SCR'. Versions D and E create another copy as WDREAD.EXE in the Windows system folder.

Magold adds a new subkey in the registry under

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\
CurrentVersion\Run
```

With value:

```
"raVe"="{Windir}\raVe.exe"
```

Thus the virus will execute automatically each time the computer is restarted.

The virus changes the value of the following registry keys:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Classes\exefile\
shell\open\command

HKEY_LOCAL_MACHINE\SOFTWARE\Classes\comfile\
shell\open\command

HKEY_LOCAL_MACHINE\SOFTWARE\Classes\batfile\
shell\open\command

HKEY_LOCAL_MACHINE\SOFTWARE\Classes\piffile\
shell\open\command

HKEY_LOCAL_MACHINE\SOFTWARE\Classes\scrfile\
shell\open\command
```

The new value of each of these keys will be

```
{WinDir}\raVe.exe "%1" %*
```

This means that the execution of EXE, COM, BAT, PIF and SCR files will pass through the virus. First the virus will execute, and then it will run the original program.

If the name of the program contains any of the following strings: VI, AV, NORTON, MCAFEE, \STARTUP, \IND, the program will be terminated immediately after execution. The STARTUP string is included in order to disable the execution of applications running from the startup folder, while IND is included for the same purpose on Hungarian localized *Windows* versions (where startup is called Inditopult).

This blocks the execution of several popular anti-virus programs. After disinfection of the virus, these registry keys should be restored to their original state, otherwise the file types listed above will not execute, which renders the incompletely disinfected systems practically useless.

NETWORK SHARE SPREAD

The worm copies itself to mapped network drives as 'Maya Gold.scr'. It also creates an autorun.inf file with the content:

```
[autorun]
open=Maya Gold.scr
```

This means that the worm will execute when the drive is opened in an *Explorer* window.

IRC SPREAD

The virus can spread via IRC. It modifies the configuration scripts script.ini of the IRC and events.ini of the pIrch clients in order to spread the virus, when other IRC users are joining the same channel as that to which the infected user is connected. It will only insert the line spreading the virus if these files already exist prior to the infection.

The entry in the configuration files will point to Maya Gold.SCR which is created in the Windows directory.

P2P SPREAD

Magold copies itself to the shared directories of the following peer-to-peer file exchange programs: *Limewire*, *Gnucleus*, *Sharezaa*, *Bearshare*, *Edonkey2000*, *Morpheus* and *Grokster*.

It also spreads via *Kazaa*, but in this case instead of copying itself into its directory, the virus modifies the shared directory under the registry key:

```
HKCU\SOFTWARE\Kazaa\Transfer\DlDir0
```

to the value

```
"%WinDir%\rave
```

This directory contains a copy of the worm, with the usual name: Maya Gold.SCR.

E-MAIL SPREAD

The virus collects email addresses from the Windows Address Book, from the files found on the computer, from all *.ht* files from the Internet cache directory (specified in the HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet \Cache\Path registry key), and from the *Outlook Express* mail folders.

The virus uses a very simple logic: anything to the left and right of an '@' is treated as an email address. This meant that, from my replication PC, copies of the virus were sent out to recipients such as W32.Alco.AB@mm and W32.Zokrim.B@mm. The addresses are stored in the Windows system directory in the file raVec.txt.

Magold uses its own SMTP engine to send out infected messages. The infected messages of variants A, B and C have the subject line:

```
Maya Gold-os kepernyokimelo!
```

And body text:

```
Tisztelt cím!
Az EROTIKA.LAP.HU nézettségének növelése
érdekében egy kis ízelítõt kíván adni
kínálatából az Internet felhasználóknak!
FIGYELEM: A 'Maya Gold.scr' nevû csatolt
állomány egy képernyõvédõ. Mint a neve
is mutatja Maya Gold pornószínésznõrõl
tartalmaz különbözõ képeket. Az
állományt ajánlott elõbb a lemezre menteni,
majd utána futtatni.
Amennyiben valami problémája, kérdése van,
írjon a következõ címre:
erotika@lap.hu
Üdvözlettel: EROTIKA.LAP.HU
```

The sender appears to be erotika@lap.hu, and the virus is attached as 'Maya Gold.scr'.

The messages describe the attachment as a screensaver showing pictures of the porn star Maya Gold, all in order to promote the website www.erotika.hu.

Magold.D disguises itself as a virus alert (warning about Magold.A) coming from a Hungarian anti-virus website (sender address info@virushirado.hu), the attachment name is 'ant_magold_v1', and its extension may be .exe, .com or .bat.

The subject of the message is randomly selected from the following list:

- Veszelyes virus terjed a neten!
- Gyorsan terjed a 'Magold'!
- Atszinezett Windows, kinyilt CD-rom meghajto! Itt a vedekezes!
- Azonnali vedekezes a magyar fereg ellen!

The body of the message is:

```
Kedves Felhasználó!
2003.05.29-én a Vírus Híradó tudomására
jutott, hogy új magyar vírus terjed az
Interneten, ami a 'Magold' nevet kapta.
A féreg rengeteg gépet fertoz és fertozött már
meg. A levélhez csatolt állomány
lefuttatásával ön megvédheti, ill. ha már
megfertozte a féreg leirthatja a magyar
vírust.
A 'Magold' terjedésének megakadályozása
érdekében arra kérjük, hogy ezt a levelet
küldje tovább barátainak, ismeroseinek,
kollégáinak!
Bovebb információ a féregrol:
http://www.virushirado.hu/virh-
virusleir.php?oid=268435538
E-Mail címünk:
info@virushirado.hu
Köszönettel: Vírus Hiradó
```

Magold.E spreads in messages using the fake sender address valovilag@rtlklub.hu, with one of the following subject lines:

- Videofelvetel Sziszi-rol!
- Sziszi a Valo Vilag-ban!
- Sziszi a zuhanyzoban!
- Sziszi a Voros Demon!

The body of the message is the following:

```
Tisztelt Cím!
Az RTL KLUB jóvoltából Ön most részt vehet egy
Internetes nyereményjátékban, ahol akár
10.000.000 Ft-ot is nyerhet.
Ehhez nem kell mást tenni, mint a levélhez
csatolt flash-videót lefuttatni (ami Sziszi-t
a Való Világ 2 sztárját mutatja be zuhanyzás
közben), majd a film végén megjelenõ
azonosítót visszaküldeni a
valovilag@rtlklub.hu címre és Ön máris játékba
került.
A sorsolás nyerteseit E-Mail-ben értesítjük
2003.06.30.-án.
Üdvözlettel: RTL KLUB - NA NÁ -
```

This message refers to one of the Hungarian *Big Brother* adaptations, and offers a video of one of the contestants under the shower. It claims that there is an ID code at the end of the video, which should be sent to the TV channel in order to win the equivalent of approximately 42,500 USD.

The Hungarian TV channel in question is currently considering taking legal action over the (as yet) unknown indirect damages caused by the virus – they received numerous complaints from users who failed to see the video after double-clicking the attachment.

All Magold variants send a notification message from the infected computer, containing the following information:

- IP address of the computer.
- Name of the infected computer.
- Username.
- *Windows* version.
- Service packs.
- Date and time of infection.
- A list of shared network resources (shared folders, printers etc.).

The message ends with the line:

```
PUNKS NOT DEAD
```

Or, in the D and E variants:

```
PUNKS NOT DEAD – dreAd –
```

Versions A, B and C send this message to email address rave_punk@freemail.hu, while D and E send it to address dread_punk@freemail.hu, presumably both belong to the virus author.

By now you should have a pretty good picture of the virus writer (keywords: age 18, male; interests: sex, punk and rave).

PAYLOAD

Magold executes several annoying payload routines randomly:

- Changes the colour of the window borders on the desktop and the taskbar to red.
- Periodically opens the CD player.
- Blocks the movement of the mouse cursor, thus the top region of the screen becomes unreachable.
- Appends the following text to the title of the foreground desktop windows: '*=:-) OFFSPRING is coOL =:-) PUNK S NOT DEAD =:-)*'.
- Sends a message to the printer, the rough translation of which is:

```
HELP ME!!

I am the printer and I would like to ask you
to talk to Windows because it is not accept-
able any more.

It keeps bugging me with stupid question and
request: "Do you have enough paper?", "Can
you print in colour?", "I would like it in
landscape mode!", "Are you ready now?"

As if it were that fast.

I hope you agree with me, that it can't go
on like this. Something has to be done!

BEST REGARDS FROM YOUR HELPFUL AND UNDER-
STANDING FRIEND: THE PRINTER

PUNK'S NOT DEAD

=:-)

=:-)
```

- Creates 2000 zero length files with the name 'raVe****' (where **** is the sequence number) in 'Document and Settings\All Users\Desktop'.

- Prevents the creation of processes with the following strings in their name: VIR, ANTI, AFEE, NORT, PROT, AV.
- Opens the web page http://www.offspring.com/.

These routines are present in version A, B and C. Versions D and E instead terminate all the processes whose window caption contains any of the following text: VIR, ANTI, AFEE, NORT, PROT or AV. Moreover, these variants terminate all processes whose filename contains any of the following text: VIR, ANTI, AFEE, NORT, PROT, AV or WINK. Then the virus terminates the following processes: MSCVB32.EXE, ISERVC.EXE, MSCCN32.EXE, WINGATE.EXE, WINEXE.EXE, WINRPC.EXE, SCAM32.EXE and SIRC32.EXE. This way the virus disables the execution of the popular anti-virus products.

CONCLUSION

It is an old piece of advice that it is useful to learn foreign languages. This is true for virus writers as well. Magold proved that even a computer virus couldn't manage nowadays without speaking English. Fortunately, as a result of the language constraints, the virus did not become widespread, otherwise the annoying payload routines and the registry changes regarding the executable types would have upset a lot more infected users.

VIRUS ANALYSIS 2

RUSSIAN DOLL

Adrian Marinescu
GeCAD Software, Romania

A few weeks ago I received a package from a fellow anti-virus researcher containing some supposedly new *Windows* viruses. Intrigued by the suggested virus name – which was the last name of one of my fellow researchers – I got my hands on some interesting pieces of code.

Having seen so many high-level language worms in the past year, it was a strange feeling when I started to dig inside a polymorphic encrypted EPO file infector. Usually, within the first few minutes you get a pretty good idea of the piece of code you're looking at – with Stepar, however, things were slightly different. I was in for several surprises during my analysis – surprises that many of us expected (but did not want to see).

Over the last few weeks I have received nine different variants of this virus – some of them appear to be debug versions (even having a debug console), others contain minor bug fixes. This description will refer to the initial variant I received, Win32/Stepar.15349.

INFECTION PROCESS

When running an infected file, the first thing Stepar does is to find the base of the Windows kernel – this address will be used to import the APIs the worm requires in order to replicate. The method Stepar uses is not very common – it assumes that the kernel is loaded at the same address as the exception handler – this works for known *Windows* systems.

Next, Stepar retrieves a large number of 'KERNEL32.DLL' APIs (69) by using a checksum on the API name and the API name length. To make emulation and debugging more difficult, the replication code is launched on a separate thread, while the main thread restores the code from the original entry point and calls the restored code.

20 more APIs from 'USER32.DLL' will be imported, as well as one from 'PSAPI.DLL' (if available). If a mutex named 'ZMX' is present in the system, Stepar assumes that another copy of itself is running – if this is the case, it waits until the mutex is no longer present and then continues the replication thread.

To make the infection process less suspicious, Stepar uses a technique that has been used by several *Windows* viruses before – it tries to find the 'EXPLORER' process in memory and to inject itself into that process – therefore, the viral code will not be listed as a separate process and the replication will pass unnoticed. Copying its body inside the 'EXPLORER' memory space and installing a custom hook

that points to the viral code does the job. When the hook receives control, it spawns a thread that executes the virus body and it unregisters the hook installed previously.

After the mutex 'ZMX' has been created to signify that the infection process is in progress, the current locale is checked – if the country is US (or any other country with country ID equal to one e.g. Dominican Republic, Jamaica, Puerto Rico, Trinidad & Tobago), and the current day plus the current month modulo 16 equals 4, Stepar's infection routine will target every file instead of only those matching '*.exe'. Interestingly, in this case the 'MZ' sign is not checked – this might result in damaged PE files (without that sign) that can be infected by Stepar.

Next, the current thread priority is set to idle and several tables used by the infection routine are initialized. This was the first surprise I found – Stepar uses a disassembler engine written by the infamous Russian virus writer nicknamed Zombie, the author of the notorious 'Win32/Zmist' virus (see *VB*, March 2001, p.6).

Another thread is spawned – this one will be responsible for the local network infection, and will be discussed later. The current thread searches for files in the current directory and attempts to infect them.

Next, Stepar selects root drives randomly from C to Z and searches for files to infect. Probably to avoid infecting itself when running the virus, the author included a way to deactivate the infection routine for a drive – if a directory named 'l0x' is present in the root of the drive, Stepar attempts to infect another drive. The maximum number of attempts is three – after that count is reached, the current thread will wait for 20 minutes and loop to the infection routine. A maximum number of 2000 files per search will be infected for each drive. Another interesting fact is that the infection routine loops forever – even though there is code present that should release the resources used by the viral routine, it is never called.

Files matching the following list of known security utilities will be skipped when infecting a file: 'avpcc.exe', 'avp32.exe', 'avpexec.exe', 'avpinst.exe', 'drweb32w.exe', 'spider.exe', 'spidernt.exe', 'avltmain.exe', 'apvxdwin.exe' and 'pavproxy.exe'.

The network infection routine imports three APIs from 'MPR.DLL' for the spreading process. Next, it enumerates network resources in an attempt to locate shares with writable files – Stepar will try to infect them in the same way it infects local files.

THE LITTLE DEBUGGER THAT DOES IT

The most interesting part of the virus is the file infection routine – it is by far the most complex routine from this virus. EPO viruses have tried to find a suitable place to insert a call to the decryptor/virus code or the decryptor itself in many ways. The most remarkable is probably Win32/Zmist which is able to disassemble and reassemble the host file, in between being able to insert its body, split into multiple pieces, into the host code. That complex approach requires huge memory resources – but probably the strongest advantage of the method is that the virus can get control via code branches that are not called directly – this is quite important from the point of view of detection, because scanning the execution flow is not sufficient to detect the virus in those samples.

Stepar uses a method that is not so complex but is very interesting. Instead of having to analyse the host file itself, why not use the debug API that *Windows* has to offer? This is done by creating the process as a 'debugged' process, while being invisible to the user. On *Windows 9x* systems the 'debugged' process will also be invisible in the process list, by calling the undocumented *Windows* API RegisterServiceProcess. After that, the newly created process will be traced in single-step mode – the disassembled engine will tell the tracer where to set the next breakpoint after the current instruction and the debugger thread will receive notification for each breakpoint that is encountered. This way, Stepar is able to walk through the host program instruction by instruction, without having any code that deals with specific Intel operation codes (besides the disassembler itself) and specific *Windows* PE file format.

A maximum of 21,760 instructions are analysed in trying to find a suitable isle of 450 bytes into which to insert the decryptor. If such a space is found, the original code is saved inside the virus body and the main infection routine is called. The process is very simple from that point – Stepar generates a decryptor, encrypts its virus body and appends it to the last section as the last 8192 bytes from the physical file. However, when I started to analyse the decryptor I had another surprise – the engine used is written by the same author as the disassembler engine. This engine, labelled 'RPME' by the author, is able to generate polymorphic code based on plain Intel code which is disassembled, morphed, permuted and then reassembled, in a similar way to that used by Win32/Simile (see *VB*, May 2002, p.4).

TEACHING AN OLD DOG NEW TRICKS

Metamorphic viruses have unusually high memory demands. Win32/Zmist required 32 Mb of memory, Win32/Simile required 'just' 3.4 Mb – Stepar needs as 'little' as half a megabyte in order to generate a new decryptor. That memory usage is basically limited by the fact that only the decryptor is metamorphic, and the size of the morphed decryptor is limited to 450 bytes.

The engine itself is able to carry out the following operations on the decryptor: change conditional jumps (by inverting the condition), change register mov operations by using the stack, find opcodes with the same functionality and use them, expand short jump instructions to the longer equivalents, expand loop instructions and add trash instructions. Also, the decryptor might be split into pieces and those pieces can be permuted inside the virtual buffer of 450 bytes.

DETECTION ISSUES

EPO viruses are a problem for conventional scanning – in order to find the virus you need to locate the start of the viral code, which is sometimes far from trivial. Usually the main payback of detection is the scanning speed, which suffers because of the extensive investigation needed by the scanner to locate the virus code.

Stepar's detection is not trivial, but fortunately, there are a number of weak points of the infection process upon which anti-virus scanners might speculate – starting with the fact that the virus body is always placed at the end of the file and breaking its weak encryption will solve the problem for now. However, an approach that searches for the decryptor inside the host code is not so difficult to implement and would be more generic should the author release improved versions of the virus.

CONCLUSION

After analysing very advanced computer viruses, many of us fear that parts of the code we have just looked at will be used by other malicious code writers. Even though there are not so many examples, we must consider that techniques that are used in simple direct-action viruses can be used in more complex creations which might get into the wild.

When the number of polymorphic viruses started to grow, approaches such as code emulation were seen as unsuitable for scanning because of the lower speed and higher demand on resources. Since the number of EPO and metamorphic viruses is increasing, now is the time to improve generic techniques to handle such complex viruses. After looking at the VB2003 conference programme (see http://www.virusbtn.com/) I look forward to seeing Frédéric Perriot's presentation 'Defeating polymorphism through code optimization', which may offer some hints about ways to fight metamorphism in the future.

Win32/Stepar family

Type:	Direct action Entry Point Obscuring polymorphic infector.
Size:	Depending on the variant, the virus code size is 14,903; 15,349; 15,350; 15,383; 15,694; 15,724; 15,879; 16,224; or 16,254 bytes.
Aliases:	Win32/Perenast, Win32/Stepan.
Infects:	Windows PE executable files.
Payload:	None.
Removal:	Delete infected files and restore them from backups.

VIRUS ANALYSIS 3

TIME TO RELAX?

Richard Marko
Eset, Slovakia

The days when macro viruses dominated virus prevalence charts are surely over. It all started with WM/Concept.A in 1995. By the end of 1999 macro viruses comprised almost 90% of all reported virus incidents. A little while later, their fame started to fade and, according to *Virus Bulletin* data, their prevalence dropped to 0.68% as of May 2003.

On 25 June 2003 I received a *Word* document that seemed, based on heuristics, to be infected by a new macro virus. A brief analysis showed that the heuristics were, indeed, correct. The texts displayed by the virus's payload suggested the name 'Relax' as an obvious choice, but we have already had experience with a couple of unrelated virus variants in the 'W97M/Relax' family. As a result, we named the virus 'W97M/Lexar.A'.

REPLICATION

The source code of W97M/Lexar.A consists of only 83 lines. Its replication mechanism is standard and straightforward. The entire code is stored within the ThisDocument class module and consists of four subroutines: RELAX2, Document_Close, GOODSub and Document_Open.

The Document_Open and Document_Close event handlers are responsible for the automatic activation of the virus, GOODSub (called from both Document_Open and Document_Close) performs the replication. RELAX2 (called only from Document_Close) contains the payload.

The GOODSub routine starts by disabling screen updates. The confirmation to save changes to the Normal template is also disabled. Then it exports the ThisDocument class module from the active project (i.e. itself) to a file named 'C:\temp.tmp' and reads the content of the file back to the keimeno string variable ('keimeno' means 'text' in Greek).

Since every exported module contains a header before the actual source code, the virus needs to get rid of it. This is accomplished by searching keimeno for the first occurrence of the ''RELAX' string (which is the first line of the virus's code) and stripping all the characters to the left of it using the Right function.

Now the ThisDocument class module of the active document and the Normal template are inspected successively. If they are not infected already (the ''RELAX' string cannot be found within their code) the lines from the keimeno variable are inserted at their beginning using the InsertLines function, and thus they become infected.

The virus will then attempt to open the file containing the active document in binary mode. If it succeeds it will set the byte at offset 35dh within the file to 1. We will address this unusual operation in more detail later. Finally, the virus deletes the 'C:\temp.tmp' file and re-enables the screen updating.

PAYLOAD

First, the RELAX2 procedure examines the current date. It will start the payload only if the day of the month is a multiple of 10 (i.e. 10th, 20th or 30th) and the month is a multiple of 4 (i.e. April, August or December). This leads to nine combinations. When the above conditions are fulfilled, the procedure will add 23 lines of code to the 'C:\Autoexec.bat' file. When it is executed (which happens automatically during the system boot on *Windows 9x* and *Me*) it will display the following text:

```
NOTE!!!
***
*****
*******
*****
***
Sometimes you must RELAX.
Please, RELAX while deleting all files in C:\
*****
*******
*****
GREECE
==================================
Press any key to continue . . .
```

Once a key is pressed the following lines are displayed:

```
All files deleted!!!
Now, you have a clean COMPUTER.
*******
*******
Press any key to continue . . .
```

It is probable that since the RELAX2 subroutine is called from Document_Close, the 23 lines of code are appended to 'autoexec.bat' multiple times. This means that after another key is pressed the first text appears again, and so on. Finally, after all the texts have been displayed the system boot continues. The good news is that no files are actually deleted, so you can relax, as long as you don't mind having a virus on your computer!

STRANGE BYTE

So far so good. Everything was relatively simple and

straightforward except for one thing – what was that strange byte the virus changed to 1?

While replicating the virus in our virus lab, I noticed two unusual facts. First, *Word* did not display its usual macro virus warning. Secondly, our macro engine refused to clean the infected document, complaining that the file format was incorrect. Both occurrences can be attributed to that single byte change.

Let's start with a short description of the *Word* document file format. The documents are mostly stored in OLE2 compound files, which can be described as a number of individual files bound together in one physical file. The complex physical file layout resembles a file system with two-level FAT, directories built as AVL trees etc. This means we deal with various storages (directories) and streams (files), which have their own internal format.

Every OLE2 compound file is a sequence of sectors, which are normally 200h bytes long. The first sector (at file offset 0) starts with a header. It contains all the important information about file layouts, such as the size of the sector, pointer to the first sector of the root directory and so on. The order of the other sectors depends on FAT content and generally is not guaranteed. Now, how can the virus write to the file offset 35dh (contained in the second sector provided the sector size is 200h) without causing uncontrolled damage?

The answer comes from the way in which *Word* documents are stored. Our testing shows that the majority have the first sector of the WordDocument stream stored at file offset 200h – right after the first file sector. Therefore, the file offset 35dh actually maps to the offset 15dh within the WordDocument stream.

WordDocument STREAM

With part of the puzzle solved, let us look more closely at the WordDocument stream. It is the fundamental stream for any *Word* document. It starts with a structure called FIB (File Information Block). Near the offset 15dh we find two interesting fields: fcCmds and lcbCmds.

Let us look at the *Microsoft* documentation:

```
0x015A fcCmds  long    offset in table stream of
                       the macro commands.
                       These commands are
                       private and undocumented.
0x015E lcbCmds ulong   undocument size of
                       undocument structure not
                       documented above.
```

It looks to me as if someone at *Microsoft* was having fun. Clearly, setting the byte at offset 15dh changes the most significant byte of the fcCmds double word. Since the table stream is often a few kilobytes long, fcCmds will point beyond the stream end.

Although the 'macro commands' area is officially undocumented, a bit of reverse engineering shows that it contains a sequence of different records. For example, they contain the names of macros in the VBA project, information on user customizations, etc. The area always starts with the byte 0ffh and ends with the byte 40h.

Documents that do not contain macros or user customizations have lcbCmds = 2 and fcCmds pointing to 0ffh, 40h. When our macro engine removes macros from an infected document it wipes off the whole 'macro commands' area.

Since W97M/Lexar.A sets fcCmds beyond the end of the table stream the engine complains and refuses to continue. Now that we have the second third of the puzzle solved it is time to answer the 'ultimate' question – why didn't *Word* warn us about potentially dangerous macros?

MACRO VIRUS PROTECTION

Up to *Office 95*, *Microsoft* used WordBasic as a language for macros in its *Word* text processor. The macros were stored within the WordDocument stream. Then, starting with *Office 97*, *Microsoft* replaced WordBasic with the more powerful VBA. The macros are now stored in the Macros storage. It is quite simple: no macros, no Macros storage.

We will now examine the function responsible for the 'macro virus protection' in *Word 97 SR-1*. The function is buried inside winword.exe, which is 5 Mb long but, using a bit of a trial-and-error approach and (some) practice, it is possible to find it. We will name it *ProcessMacroCommands*. Its partial and simplified implementation follows:

```
BOOL ProcessMacroCommands(STREAM_HANDLE
TableStream, LONG fcCmds, ULONG lcbCmds, …)
{
        BYTE RecordID;
        BOOL UserWarned=FALSE;
        if (lcbCmds == 1) return FALSE;
        …
        Seek(TableStream, fcCmds);
        …
        ReadByte(TableStream, &RecordID);
        lcbCmds–;
        if (RecordID != 0xff) return TRUE;
        ReadByte(TableStream, &RecordID);
        lcbCmds–;
        if (RecordID != 0x40)
        {
PossibleMacros:
                if (!WarnUserAboutMacros())
                        return FALSE;
```

```
                UserWarned=TRUE;
        }
        if (!UserWarned && TestMacrosStorage())
            goto PossibleMacros;
        …
}
```

The macros are enabled by default. If the function returns FALSE, *Word* will not open the document. The ReadByte function reads one byte from the specified stream to the selected location. The offset in the stream is set by the Seek function. If it is set beyond the end of the stream, ReadByte will read 0.

The WarnUserAboutMacros function displays a well-known dialog (if it is enabled in the *Word* configuration) and, depending on the user selection, will either enable or disable macros (by setting some internal flag) or even return FALSE if the user opted not to open the document. Finally, the TestMacrosStorage function tries to open the Macros storage and returns TRUE if it succeeds.

As can be seen, once fcCmds is set beyond the end of the stream the comparison to 0ffh will fail and the function will return TRUE (i.e. *Word* continues opening the document) without displaying the warning prompt or disabling the macros. The puzzle is now solved. Clearly, the same effect could be achieved by setting fcCmds to any stream offset that does not contain 0ffh in the first byte or by overwriting the first byte of the 'macro commands' area.

SOLUTION

There is good news and bad news. The bad news is that this vulnerability also exists in *Word 2000* and *Word XP*, even if the Security Level for macros is set to High. Macintosh versions of *Word* are probably affected too, but we have not tested them. The good news is that *Microsoft* has known about the problem since 21 June 2001 and described the vulnerability in *Microsoft Security Bulletin MS01-034*. The patches for different *Word* versions are contained therein.

If you are curious (as I was) to see how the patch affects the ProcessMacroCommands function, keep reading. Here goes:

```
BOOL ProcessMacroCommands(STREAM_HANDLE
TableStream, LONG fcCmds, ULONG lcbCmds, …)
{
        BYTE RecordID;
        BOOL UserWarned=FALSE;
        if (lcbCmds == 1) return FALSE;
        …
        Seek(TableStream, fcCmds);
        …
        if (TestMacrosStorage())
        {
                If (!WarnUserAboutMacros())
                        return FALSE;
                UserWarned=TRUE;
        }
        ReadByte(TableStream, &RecordID);
        lcbCmds–;
        If (RecordID != 0xff) return TRUE;
        …
}
```

As illustrated above, another test for the Macros storage was inserted before the actual processing of the 'macro commands' area begins. Since this storage is present whenever VBA macros are included in a *Word* document, the protection is more secure now and the trick used by W97M/Lexar.A no longer works.

CONCLUSION

W97M/Lexar.A is a relatively simple macro virus. The trivial way in which it bypasses macro virus protection of the *Microsoft Word* text processors makes it quite interesting.

The fact that it changes a byte at a fixed file offset means that some documents can become corrupted. Cleaning of such documents in general can prove problematic. Even if it changes the desired byte in the ThisDocument stream, the cleaning algorithms of anti-virus products should be able to handle and correct it. That might, eventually, require certain macro engine adjustments. Recently I noticed that an intended variant of W97M/Lexar.A has been known since November 2002. That variant is capable of infecting the Normal template but is unable to spread further due to a bug in its GOODSub subroutine.

It can be expected that, even though the problem has been known for two years now, the majority of *Word* installations are still vulnerable (and are likely to remain so). The behavioural pattern of a 'generic' end-user and the nature of the OSs and applications design presents a constant challenge to all AV vendors to protect a user who couldn't care less.

W97M/Lexar.A

Aliases:	W97M/Xaler.B, W97M/Relax.
Type:	*Word* macro virus.
Payload:	Displays text on system start.
Self-recognition:	''RELAX' string within the code.
Removal:	Restore infected files from backup or try to use a virus scanner.

FEATURE

FLOODING FROM THE UNDERGROUND – A GLOBAL THREAT

Scott Molenkamp
Computer Associates, Australia

When Khaled Mardam-Bey developed an IRC client for the *Windows* platform, I doubt he envisaged *mIRC* becoming the basis for the control of an immeasurable number of compromised machines in bot-nets. Khaled has the original authors of the Global-Threat (GT) bot to thank for that.

The original GT bot exploited *mIRC*'s powerful scripting language, which included support for raw socket connections, to create a bot that was easily controllable via IRC. Functionality included, but was not limited to, port scanning, packet flooding, and provision of BouNCe.

The bot was open source (owing to *mIRC*'s interpreted scripting) and easily configured. This attribute has allowed it to spawn a myriad of variants, many of which provide completely new functionality.

While the problem is currently understated and difficult to quantify, the trend is definitely on the up. Currently these pervasive bots are displaying worm-like ability and exploiting flaws in *Windows* or weak security. The two authors of one such variant were arrested in February 2003, with the UK's National Hi-Tech Crime Unit reporting that over 18,000 computers had been infected.

BACKGROUND

IRC stands for Internet Relay Chat. It provides a way in which people can chat to each other over a network in real time. The people who wish to chat to one another run a client on their machine and connect to an IRC server.

Jarkko Oikarinen originally created IRC in 1988, planning for a maximum concurrent user base of around 100. The IRC protocol was later defined by RFC1459 in 1993. It has since been updated to include RFC2810, RFC2811, RFC2812 and RFC2813 (see http://www.rfc-editor.org/).

Generally speaking, an IRC bot is a non-human client with programmed responses to various events. A bot can be used for all kinds of useful purposes, such as granting operator status to recognised users.

These days a bot is more likely to be referred to as something that is associated with more nefarious activities. When these bots are gathered together under the control of a common overseer they are often referred to as a bot-net. IRC is utilised as the communication medium between the overseer and the bot-net, sending commands either individually or en masse.

IRC, BACKDOORS AND ME

Originally, *mIRC* was not very popular for use as a backdoor control mechanism. The obvious drawback is that the user must actually be running *mIRC* in order for a malicious script to be active.

Also, the original concept was to 'backdoor' an existing *mIRC* client by installing script files that contained the desired functionality. Many IRC worms heavily exploited an original design flaw in *mIRC* that created downloaded files in the same directory as the program, thus allowing them to overwrite configuration file 'script.ini', which is loaded automatically by *mIRC*. This flaw was rectified in *mIRC 5.3* (December 1997).

Some of these worms did have rudimentary backdoor-like control, mostly related to IRC functions. One of these worms, IRC/Jobbo, implemented more expansive commands, such as the ability to run local files. *mIRC* was not the only client able to be exploited. A popular UNIX client, *IRCII*, was also used to interpret scripts with backdoor functionality.

One of the events which may have led to IRC being perceived as a useful control medium for backdoors was the release of *SubSeven 2.1* in November 1999. It permitted a *SubSeven* server to be controlled via a bot connected to an IRC server.

This method of control is typical of an IRC backdoor and is displayed in the diagram shown in Figure 1. IRC messages travel to and from the IRC server to the clients (represented by the green lines). Control messages from the overseer travel to and from the bot either directly or via the IRC server (represented by the blue lines).

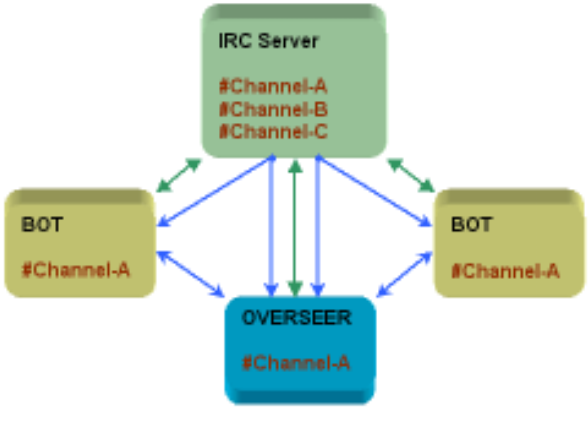


Figure 1

At the end of October 2000, when a compromised user posted a message to a security mailing list complete with a fully functional GT bot, it was plain to see that *mIRC* was now being used as the Trojan engine in its own right.

MIRC SCRIPTING

At the heart of each of these bots is the popular shareware client for *Windows*, *mIRC*. More specifically, it is *mIRC*'s scripting functionality that is utilised by each bot to determine its actions.

mIRC has the ability to interpret 'scripts that react to IRC server events', which are referred to as 'remotes'. Equally important for functionality is *mIRC*'s support for raw socket connections. By the release of *mIRC version 5.5* (January 1999), support for both TCP and UDP had been implemented.

One of the fundamental ways in which *mIRC* acts with regard to events is related to access levels. Each event is given a level. In addition, each user is given an access level, the default for which is 1. This assignment facilitates the ability both to restrict and to allow different users to trigger different events.

Probably the most important event for control is when a message is received by the client. This can be a private message, a channel message, or both. This event is referred to as 'TEXT'.

The syntax for acting upon such an event is:

```
on <level> : TEXT : <pattern> :
<messagesource> : <commands>
```

where

- *level* is the minimum level required to access the event
- *pattern* is the text which will trigger the event
- *messagesource* is where the message originated from (private, channel, or either).

For example, the following events will be triggered:

```
on * : TEXT : !hello : * : { commands }
```

the message !hello is received from any source from a user regardless of their level.

```
on 10 :TEXT : !goodbye* : # : { commands }
```

the message beginning with !goodbye is received via a channel from a level 10 (or above) user.

Often these are broken down even further. In the following example the first line ensures that commands within the first set of braces are executed only if a message is from a level 10 (or above) user. The second line checks the equivalence of the first parameter.

```
on 10 :TEXT : * : * : {
     if ($1 == !exit) { commands }
}
```

There are other common events that are used in *mIRC* scripting, such as CONNECT and START. The CONNECT event is triggered when a connection to an IRC server is made. The START event is triggered when a script is loaded.

Their usage may appear as:

```
on * : CONNECT : { }
on * : START : { }
```

MIRC SOCKETS

To access *mIRC*'s raw sockets functionality, knowledge of only a few simple commands is required. These commands are:

SOCKOPEN

SOCKCLOSE

SOCKREAD

SOCKWRITE

SOCKLISTEN

SOCKACCEPT

SOCKUDP

Barring SOCKACCEPT, each of these commands has an associated event that can be triggered. The name of the event is the same as that of the command, with the exception of SOCKUDP, whose corresponding event is named UDPREAD.

For example, an http download script may contain the following code:

```
SOCKOPEN httpsock www.myhost.com 80
on * : SOCKOPEN : httpsock : { SOCKWRITE -n
$sockname GET / HTTP/1.0 }
```

The SOCKOPEN event is triggered when a successful connection to www.myhost.com is made. The –n tells SOCKWRITE to append a carriage return/line feed to the end of the data sent. $sockname is a *mIRC* identifier, in this case it is httpsock.

The SOCKCLOSE event is triggered when the remote host closes the connection. The SOCKLISTEN event is triggered when an inbound connection is made to a port.

FINAL WORD

There are many other scripting capabilities that are utilised by bots. These include timers, which allow commands to be

executed repeatedly with a specified delay. *mIRC* has many file and string manipulation functions, such as regular expressions and tokenizers.

When reading *mIRC* scripts it is worth noting that variables are prefaced with %. An identifier, $, returns the value of a variable, whether it is a *mIRC* variable such as $sockname or one a script has created for its own use.

mIRC also allows other operating system interaction such as the ability to execute local files and make both DLL and COM object calls. Though, strangely enough, the usual suspects Scripting.FileSystemObject and WScript.Shell never seem to be called upon.

BOT FUNCTIONALITY

Most GT bots provide widely varying functionality. Typically they all share at least some of the basic functionality of the original, with a few extras.

The original had the ability to provide:

- Bounce (BNC): A BNC is a method by which you use a machine other than your own as a gateway to an IRC server. This is not necessarily a malicious activity in itself. Using a BNC enables a user to protect themselves from Denial of Service attacks, as their client IP address becomes masked by that of the BNC provider.
- Port Scanning: The ability to test for open ports over a given IP range.
- Cloning: A clone is the term given to any connection from the same source over and above the first connection to an IRC server. Clones can be loaded via open gateways such as BNC and can be used to flood an IRC server/channel.
- Flooding: Whether implemented through the use of *mIRC* scripting or by calling out to various standalone ICMP/IGMP/UDP flooders.

WILLIAM TELL'S WATERMELON

The target hosts for GT bots are *Windows*-based machines, given their obvious dependency on *mIRC*.

The original GT bot had no self-spreading capabilities – instead, social engineering methods were used to entice users to download installers.

One of the well-known 'inviter' messages read as follows.

```
:!Notice!: A Recent Port Scan on your Computer
reveals that Port 1800 is in open state. This
usually means that you have been infected with
an IRC Worm Virus. Please download the cleaner
at:http://www.No-Hack.Us/Fixes/Worm1800.exe to
remove the virus from your system. If you do
not comply with this rule within 30 minutes,
our client monitor will ban you from this
network.-Thanks For Understanding. UNDERNet
Exploit Team
```

Usually, this single executable is either a downloader set to retrieve an installer package, or else the installer package itself.

The installer contains all of the necessary components of the backdoor. At the very least, this would include a copy of *mIRC*, a malicious script file (probably named 'mirc.ini') and a program to hide the *mIRC* GUI window from the user. At the other end of the scale, dozens of files could be included, with many scripts, command line utilities, flooders, Dynamic Link Libraries and servers of various types.

These packages are normally created with freely available installers which allow the components to be silently installed and executed. Those commonly used are Setup Factory, Instyler, Install Wizard, PaquetBuilder, GSFx Wizard, NSIS, SFXMaker and RARSFx.

EASY TO SPREAD AND BETTER THAN BUTTER

Part of the evolution of the GT bot was to include methods for automatically spreading to other machines in a worm-like fashion.

Some lesser-used methods of spreading include exploiting various Trojan protocols such as *SubSeven* or *NetDevil* to upload an installer onto a machine. Old exploits such as *IIS Web Server Folder Traversal* (MS00-078) are still employed to great effect.

The most infamous Trojan to use the *IIS Web Server Folder Traversal exploit* was TKBot, the authors of which were arrested earlier this year.

The most common method of autospreading is via the SMB (Server Message Block) protocol. SMB can be used over multiple protocols such as TCP/IP and NetBIOS.

It is the application of the SMB protocol in *Windows* file sharing which is abused. This is achieved through the use of the *Sysinternals* freeware utility *PsExec,* which allows a user to execute any process on a remote system. However, it is the presence of weak or obvious passwords on user accounts that allows the use of *PsExec* to be a viable means of attack in the first place – see Martin Overton's 'You are the weakest link, goodbye! – Passwords, malware and you' (*VB*, July 2003, p.12).

However, it is probably the rise of *NT*-based operating systems such as *Windows XP,* especially in the home

environment, which is fuelling the further rise in prevalence of these bots.

CONSTITUENTS AND MOTIVATION

The constituents of a GT bot will differ depending on the desired functionality of the bot. The driving force behind the use of GT bots is the ability to gain control of a resource, whether that is diskspace, bandwidth, anonymity or any other tangible benefit an evil-doer could see as an advantage.

GT-styled bots are now being used to control compromised machines which are part of public warez/pornography distribution networks, or pubstros. Included in the scripts are triggers that allow the pubstro machine to be easily 'administered'. These bots will include a copy of a popular FTP server such as Serv-U or SlimFTPd.

Alternatively, a bot on a compromised machine may be geared to the more traditional functions of packeting and port scanning – in which case the package is likely to include compiled flooders and possibly an IRC server. In fact, many of the packages install clean software and non-malicious components.

PROBLEMIRCTIC

With the presence of numerous innocuous elements, vigilance is required by the anti-virus industry to ensure that care is taken when adding detection. This is particularly pertinent with regard to the *mIRC* client itself.

Given the widespread use of *mIRC*, there is potential for false-alerting users to the presence of a so-called Trojan. Renaming is not sufficient to trigger detection, as a power user may have made a copy of *mIRC* so as to be able to make simultaneous connections to different IRC networks. Packing takes it one step closer; modifying the client itself certainly seems to be a boundary at which some anti-virus companies have drawn the line.

There are multiple reasons why a *mIRC* client may be modified. First, the default configuration file mirc.ini is created if *mIRC* does not find it in the current directory. Usually the name of this file is modified so that *mIRC* looks for a file with a custom name. The reason for doing this is to disguise the presence of a rogue *mIRC* in what would be an unexpected location on a machine.

GT bots are often installed to 'special' *Windows* directories such as \windows\fonts, \windows\inf and \winnt\system32\catroot\ to hide their presence. Other modifications include the diskspace saving resource removal and cleverly misdirected registry key creation.

Owing to the open source nature and widely ranging functionality of this particular variety of IRC bot, the number of variants is immense. Merely categorising the bots is not an easy task: developing a logical naming scheme for these bots is not really possible.

The open source nature of these bots also enables scripts to be reused, rearranged, removed, added, split and modified in countless ways. There may exist a single package where all scripts are detected, but all with completely different variant names. Various files within the bot package usually have wide-ranging 'platform' prefixes, such as Win32, BAT, VBS, REG and IRC.

Confusing for the user who has had the misfortune to have their machine compromised.

IRC THE FUTURE

The prevalence of these GT-styled bots is increasing. Whilst being unable to quantify the exact number of compromised machines, it would be safe to say that there are probably hundreds of bot-nets currently in use, if not more.

Taking an example of the 18,000 strong bot-net that was uncovered earlier this year, we could calculate a conservative possible bandwidth of:

```
56kbit * 18,000 = 984Mbit
```

This magnitude of bandwidth could easily be used to disrupt the service of both IRC and web servers alike.

The delivery of these bots in the future may be melded with rootkits, as well as with firewall bypass/removal functionality.

RECOMMENDATION

There are a few steps that can be taken to avoid your machine being compromised.

- First, make sure that both *Windows* and your anti-virus protection are up to date with the latest patches/signatures.
- Secondly, make sure that all accounts have suitable passwords.
- Finally, install a desktop firewall that can block the outbound connections made to IRC servers that the bots require for control. These are usually on ports 6000–6669, but may be different.

If you are an IRC user make sure that outbound IRC connections can only be made to specified hosts. In addition, the *Microsoft Baseline Security Analyser* can be employed to report common system misconfigurations.

COMPARATIVE REVIEW

NETWARE 6.0

Matt Ham

Preparations for a comparative review are, by now, a relatively automated response here at *Virus Bulletin*: check WildList, check product patches, check last year's notes and so the list goes on. It is at the checking of patches stage during *NetWare* comparatives that waves of unwelcome memories come flooding back – of vast patches, servers abending and long hours spent cursing the strangely poised folk whose images emblazon the OS. The notebook reading stage heightens this sense of foreboding, with strange errors and even stranger workarounds peppering last year's text. By the end of the preparations for this *NetWare* review thoughts of impending doom were greatest in my mind.

The version of the OS used in this test was *NetWare 6.0* with service pack 3 – the service pack being the usual several hundred megabytes in size. Installation of the patch failed to produce any problems, resulting in a patched and running server within minutes of beginning the process. With this promising start, the outlook seemed brighter and test sets and products could be considered.

The test set used was derived from the May 2003 WildList and, as expected, there were a large number of new worms to add to the collection. Inspection of these during replication led to the conclusion that none of the newcomers were destined to be tricky to detect – non-polymorphic worms not being the most challenging files for a scanning engine. At this stage the review looked set for a bumper crop of VB 100 % awards.

As for the products submitted for the review, there were a total of 11. In last year's *NetWare* review (see *VB*, August 2002, p.17) only nine products were on offer, so where do the differences lie? Out of the running is the now doomed *RAV for NetWare*, shelved after *GeCAD*'s takeover by *Microsoft*. This left three new arrivals, which were products from *Symantec*, *Command* and *Computer Associates*' US-based division. These products certainly existed at the time of last year's review, but were not available in a tested form for *NetWare 6*.

CA InoculateIT 4.5

ItW File	100.00%	**Macro**	99.90%
ItW File (o/a)	100.00%	**Macro (o/a)**	99.90%
Standard	99.82%	**Polymorphic**	99.89%

Although the rebranding of this product to its new title *eTrust AntiVirus* has reached the product packaging, within the documentation and internal references the name *InoculateIT* is still predominant, hence the choice of name in this review. In some places the even older product name *InocuLAN* is mentioned, so it is to be expected that it will be a long time before the current name change takes full effect.

The first problems with this product arose upon installation, with the installer declaring that it would only run on *Windows 3.x* systems – certainly an odd statement. Ignoring this error, the *InoculateIT* NLMs and associated files were installed to the server by use of this client-side application. A few patches and updates were then applied manually before the server was rebooted to make sure of a full upgrade process.

From this point onwards the testing process proceeded smoothly, though there were a few surprises. For one, the rate of scanning for clean files was among the slower of those products reviewed. More surprisingly the clean executable set was the source of two false positives. While detection was good, W97M/Pain.A was a surprise miss, the two false positives were sufficient to deny *InoculateIT* a VB 100% for the first time in many months.

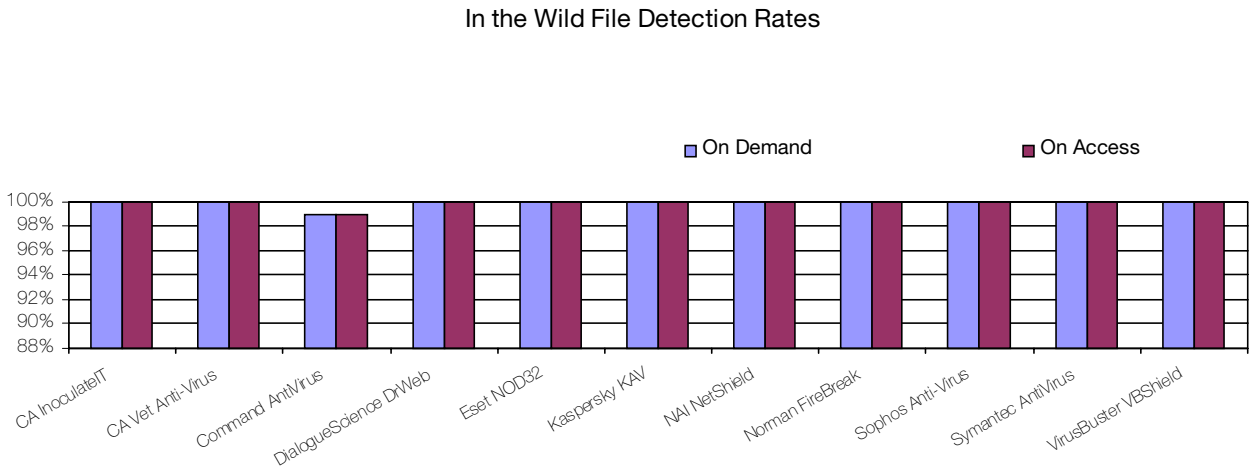
CA Vet NetWare AntiVirus 10.5.8

ItW File	100.00%	**Macro**	99.82%
ItW File (o/a)	100.00%	**Macro (o/a)**	99.82%
Standard	99.90%	**Polymorphic**	98.50%

With its sister brand suffering a little in this review, the performance of *CA Vet* was of more interest than usual. Installation proceeded with no problems at all, being performed by a client-side *Windows* application. Update files were copied manually to the SYS:\system directory from where they update the application automatically. This method of updating is notable for the fact that the *Vet* updates are a collection of files rather than one monolithic object. The update process is triggered through one and one only of these files, making it imperative that this be the last of the files copied to the server.

On to the operation of *Vet*, which was overall a slightly less taxing experience than that of its sister product, with faster scanning in the clean sets and no false positives generated. Scanning of the clean sets was an issue for the zipped OLE files, however, where the rate of scanning was far slower than expected in comparison with other clean set scans. However, the problem was not fatal and *CA*'s *Vet NetWare AntiVirus* can therefore claim the first VB 100% award of this review. Misses in the test sets were here identical with those seen in other *Vet* products on other platforms.

Another irritation in *Vet*, common to several of the products, was that only one on-demand scan may be saved at any one



time. This makes the scanning of different areas at different times much more of a chore than might otherwise be the case.

Command AntiVirus 4.70.0.20710

ItW File	98.96%	**Macro**	100.00%
ItW File (o/a)	98.96%	**Macro (o/a)**	100.00%
Standard	98.96%	**Polymorphic**	100.00%

Command AntiVirus was supplied for this review in the form of three archives. One of these is a *Windows* application which installs the client and server side consoles, the other two contain engine and update components. Neither of the consoles were particularly user-friendly, the greatest initial problem being that it seemed impossible to tell whether a scan was actually in operation from the *Windows* version.

Further use of the consoles only added to the sense of frustration since, once applied, settings did not necessarily seem to be implemented. Examples of this behaviour included files being blocked on access when the on-access component was totally disabled and files being renamed and quarantined when set for deletion.

As a result of these quirks the scanner was tested by setting the delete option, repeatedly scanning the test set and deleting renamed files. This process was continued until no more files were flagged as infected.

The resulting figures showed that detection was good with the exception of one category. That category was .HTM-extensioned files, where none of those present in the test set were detected either on access or on demand. This alone was sufficient to deny *Command AntiVirus* a VB 100% award. These misses of .HTM files occurred despite '.HT*' being included on the list of extensions to be scanned.

DialogueScience Dr.Web 4.29c

ItW File	100.00%	**Macro**	100.00%
ItW File (o/a)	100.00%	**Macro (o/a)**	100.00%
Standard	100.00%	**Polymorphic**	100.00%

Dr.Web is the first product in this review which relies entirely upon manual placing of files as its installation method. It also has one of the more basic-looking console views, the GUI being in shades of luminous green which would not have looked out of place on an early *Commodore*.

However, the lack of what could be termed as mod-cons does not detract from the functionality or effectiveness of the product.

Scanning speeds were good and only the usual *Dr.Web* warnings of possible infection were present, rather than any full-blown false positives.

As far as the interface was concerned, only one irritation stood out. This was the fact that on-demand scans did not exist on the interface. This might, at first, seem to be a serious omission, but is in fact much less important than it might seem. All that is required to operate an effective on-demand scan is to produce a scheduled scan one minute or so in the future. The ability simply to set up a scan to operate 'now' would be a welcome addition, though.

Detection rates for *DialogueScience*'s offering returned to their usual high levels after a recent blip in previous reviews, and full detection was recorded on demand. On access there were misses of samples within .ZIP and .EML

files, though nothing sufficient to deny *Dr.Web* another VB 100% award

Eset NOD32 1.455(20030707)

ItW File	100.00%	**Macro**	100.00%
ItW File (o/a)	100.00%	**Macro (o/a)**	100.00%
Standard	100.00%	**Polymorphic**	100.00%

NOD32 is another product which is installed manually by copying files directly to the server. It is also unusual in having two scanning NLMs, both of which operate through command line parameters and do not have an interface during the process of operation.

Detection rates were good, both on access and on demand, but there were some discrepancies in the documentation when the help options were triggered at the command line. These declared that the scanning of archives was set as off by default – quite at odds with the detection of W32/Heidi.A within .ZIP files. Such misleading documentation, however, is not sufficient to cause any major commotion.

With the aforementioned full detection of infected files, there were also no false positives noted in the clean sets. *NOD32* thus receives a VB 100% award in this review.

Kaspersky AntiVirus 4.00.07

ItW File	100.00%	**Macro**	100.00%
ItW File (o/a)	100.00%	**Macro (o/a)**	100.00%
Standard	100.00%	**Polymorphic**	99.92%

Kaspersky's anti-virus products for *NetWare* have traditionally been among the better integrated into the operating system, and this offering was no disappointment on that front. Installation was quite a lengthy process in comparison with some, though this resulted in a console which is integrated into *NetWare*'s ConsoleOne interface. A further *Windows*-resident interface is also installed. The combination of these two control possibilities maximises the possibilities for control within a GUI and avoids some of the more irritating aspects of *NetWare*'s classic interface look and feel.

No false positives were recorded during the clean set testing and there were few misses in the infected samples. On demand a single sample of W32/Etap was missed, while only the .ZIP samples of W32/Heidi.A were additional misses on access. With such a performance KAV is duly awarded a VB 100%.

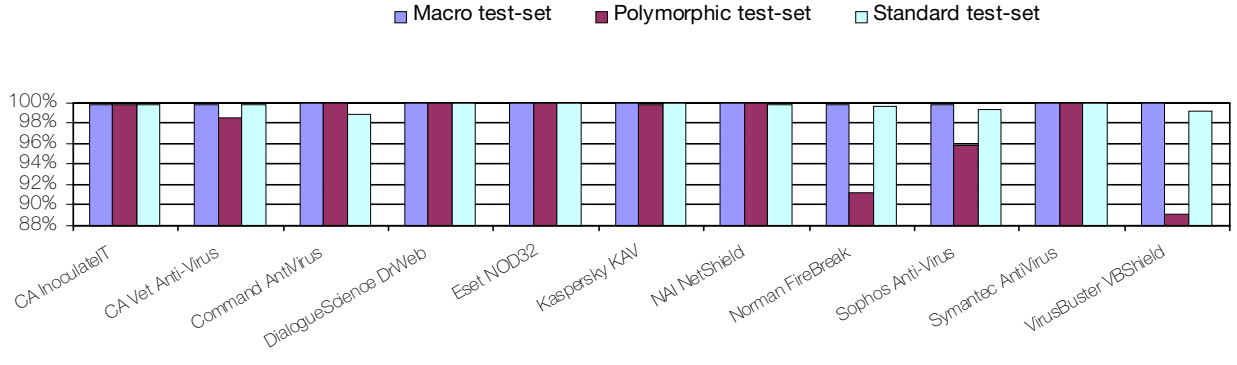
NAI McAfee NetShield 4.61 4.2.40 4.0.4275

ItW File	100.00%	**Macro**	100.00%
ItW File (o/a)	100.00%	**Macro (o/a)**	100.00%
Standard	99.82%	**Polymorphic**	100.00%

NetShield is controlled and installed through a *Windows* application, the installation of which requires, in turn, the installation of the Java Runtime Environment. Updates were performed on this occasion through unloading the application and inserting the required files. Upon reloading, the update occurred.

As was the case with several products on offer, scanning was slower than would seem comfortable. This was noted especially where files infected with W32/CTX.A were concerned, though the scanning rate of the clean set was also somewhat on the slow side. A further irritation

Detection Rates for On-Demand Scanning

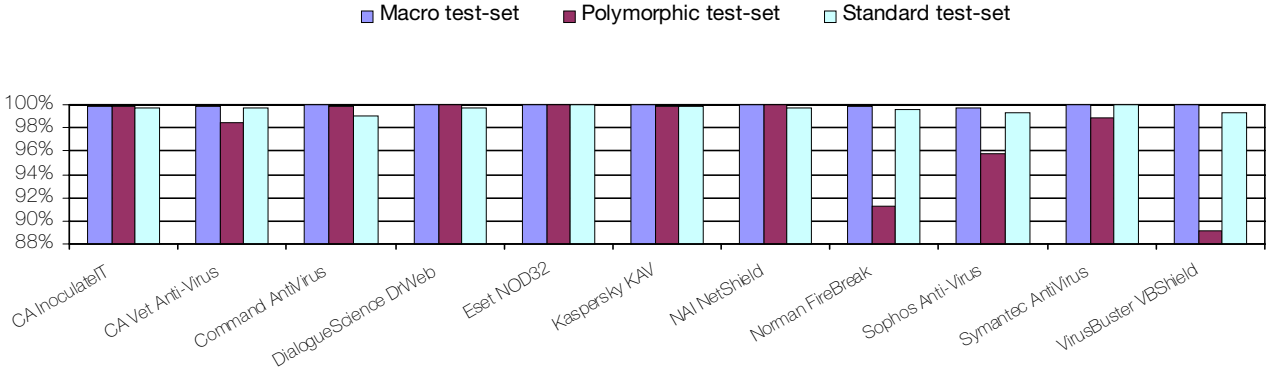


On-access tests	ItW File		Macro		Polymorphic		Standard	
	Number missed	%	Number missed	%	Number missed	%	Number missed	%
CA InoculateIT	0	100.00%	4	99.90%	1	99.89%	3	99.70%
CA Vet NetWare AntiVirus	0	100.00%	12	99.82%	437	98.50%	4	99.78%
Command AntiVirus	5	98.96%	0	100.00%	2	99.91%	14	98.96%
DialogueScience Dr.Web	0	100.00%	0	100.00%	0	100.00%	3	99.70%
Eset NOD32	0	100.00%	0	100.00%	0	100.00%	0	100.00%
Kaspersky KAV	0	100.00%	0	100.00%	1	99.92%	2	99.88%
NAI McAfee NetShield	0	100.00%	0	100.00%	0	100.00%	4	99.70%
Norman FireBreak	0	100.00%	4	99.90%	180	91.24%	11	99.64%
Sophos Anti-Virus	0	100.00%	11	99.73%	60	95.79%	15	99.31%
Symantec AntiVirus	0	100.00%	0	100.00%	35	98.86%	0	100.00%
VirusBuster VBShield	0	100.00%	0	100.00%	599	89.14%	13	99.34%

– though not confined to *NetShield* – was the inflated count of scanned files, which included all files within self-extracting archives in the count of 'files scanned'. For all this, however, no false positives were encountered.

Misses were limited in number – with samples of JS/Unicle and W32/Heidi.A comprising the total number of infections that were undetected. With this performance, therefore, *NetShield* is deserving of a VB 100% award.

Detection Rates for On-Access Scanning



On-demand tests	ItW File		Macro		Polymorphic		Standard	
	Number missed	%	Number missed	%	Number missed	%	Number missed	%
CA InoculateIT	0	100.00%	4	99.90%	1	99.89%	1	99.82%
CA Vet NetWare AntiVirus	0	100.00%	12	99.82%	437	98.50%	2	99.90%
Command AntiVirus	5	98.96%	0	100.00%	0	100.00%	14	98.96%
DialogueScience Dr.Web	0	100.00%	0	100.00%	0	100.00%	0	100.00%
Eset NOD32	0	100.00%	0	100.00%	0	100.00%	0	100.00%
Kaspersky KAV	0	100.00%	0	100.00%	1	99.92%	0	100.00%
NAI McAfee NetShield	0	100.00%	0	100.00%	0	100.00%	2	99.82%
Norman FireBreak	0	100.00%	4	99.90%	180	91.24%	11	99.64%
Sophos Anti-Virus	0	100.00%	8	99.80%	60	95.79%	13	99.40%
Symantec AntiVirus	0	100.00%	0	100.00%	0	100.00%	0	100.00%
VirusBuster VBShield	0	100.00%	0	100.00%	600	89.13%	14	99.15%

Norman FireBreak 4.60.2211

ItW File	100.00%	**Macro**	99.90%
ItW File (o/a)	100.00%	**Macro (o/a)**	99.90%
Standard	99.64%	**Polymorphic**	91.24%

Firebreak demonstrates another method of installation to add to those encountered already. The *Windows* GUI installer launches a *Windows* ConsoleOne view part-way through the installation process, which must be tweaked a little before the installation can be completed. HTML help files are opened during this process to explain in detail, if required, what steps must be taken.

Control over the installed product is primarily via ConsoleOne, whether running on a client or a server. The level of control offered is similar to that offered by most GUI virus scanners, though will be less familiar to *Norman* users, the usual *Norman* interface being far from similar to the majority.

Some oddities are present in the interface, however. It is possible to set the scanner to report only on virus detections, but this appears to trigger the conditional 'actions to be taken if disinfection fails'.

It was deemed necessary to check on-access scanning by deletion, since deletion was unavoidable in these circumstances. There was further confusion at this point, since, when copying files, the target was noted as having been deleted – whereas in fact it was the source file that had undergone this fate.

Despite these strange occurrences, all was well on the detection front. Having generated no false positives and only the usual set of missed detections, *Norman*'s *FireBreak* is worthy of a VB 100% award.

Sophos Anti-Virus 3.71

ItW File	100.00%	**Macro**	99.80%
ItW File (o/a)	100.00%	**Macro (o/a)**	99.73%
Standard	99.40%	**Polymorphic**	95.79%

Hard Disk Scan Rate	Executables			OLE Files			Zipped Executables		Zipped OLE Files	
	Time (s)	Throughput (MB/s)	FPs [susp]	Time(s)	Throughput (MB/s)	FPs [susp]	Time (s)	Throughput (MB/s)	Time(s)	Throughput (MB/s)
CA InoculateIT	751	728.3	2	61	1300.6		292	545.9	59	1264.5
CA Vet NetWare AntiVirus	177	3090.0		12	6611.1		71	2245.3	161	463.4
Command AntiVirus	1667	328.1		87	911.9		-	-	-	-
DialogueScience Dr.Web	188	2909.2	[12]	13	6102.6		77	2070.3	14	5329.1
Eset NOD32	73	7492.2		7	11333.4		108	1476.1	13	5739.0
Kaspersky KAV	315	1736.3		28	2833.3		162	984.1	44	1695.6
NAI McAfee NetShield	581	941.4		35	2266.7		166	960.3	50	1492.1
Norman FireBreak	377	1450.7		11	7212.2		27	5904.3	6	12434.6
Sophos Anti-Virus	166	3294.8		21	3777.8		40	3985.4	10	7460.7
Symantec AntiVirus	155	3528.6		24	3305.6		76	2097.6	25	2984.3
VirusBuster VBShield	234	2337.3		13	6102.6		181	880.8	18	4144.9

Sophos Anti-Virus was unique amongst products in this review in its installation method. The package is supplied as a single NLM which, when loaded, installed all the required files from within itself. This method of installation is an interesting halfway house between the two camps of total automation and manual file copying.

Other parts of the interface are, however, more irritating. It is still necessary to select all files for scanning when the target of a scan is a directory – only volumes may be scanned according to the inbuilt extension list. It is also very difficult to tell whether IDE files have been loaded so as to extend the detection abilities of the product.

Installation and interface comments aside, the *Sophos* product performed well. No false positives were apparent in any test set, and the infected samples in the ItW test set were all detected, thus earning *Sophos AntiVirus* (*SAV*) a VB 100% award. Detection rates for *SAV* are good in general, though several infected files have been missed for many months. These misses still include all the .MDB files in the test sets, although rumour has it that this detection at least will be implemented soon.

Symantec AntiVirus 8.00.0.9374

ItW File	100.00%	**Macro**	100.00%
ItW File (o/a)	100.00%	**Macro (o/a)**	100.00%
Standard	100.00%	**Polymorphic**	100.00%

Symantec AntiVirus, the second '*SAV*' in this review, is certainly the product with the most involved installation process.

In order to be installed and administered this *SAV* required *Microsoft Management Console* with the *Symantec System Center Snapin*, which requires *Internet Explorer 5* or better. Even after all these are installed, it is necessary to load the NLMs manually when first installing to a server.

This rigmarole suffices to produce an interface which is identical in look, feel and most functionality to the rest of the *Symantec* product range. This is certainly worthwhile in a large organisation, for all the added time involved when installing one server for review purposes. The process was also easier than my memories of the same process in the last *NetWare* review.

The interface being the same as other *Symantec* products, it was to be hoped that the detection rates stayed the same too. This hope proved justified, as the product showed full detection of all samples in the test sets on demand. On access, however, several samples of W32/CTX and W95/SK.8044 were missed. These samples were scanned at a noticeably slow rate on demand and on access and it seems likely that the on-access scanner is timing out while processing the files.

However, the problem files did not fall in the ItW test set, and with no false positives generated a VB 100% award is owing to *SAV*.

VirusBuster VBShield 1.17

ItW File	100.00%	**Macro**	100.00%
ItW File (o/a)	100.00%	**Macro (o/a)**	100.00%
Standard	99.15%	**Polymorphic**	89.13%

Last on the list for this month's review is *VirusBuster*'s product. This is most notable in that there are far fewer comments to be made on this occasion than in the previous review. The reason is the remarkable improvement in ease of use and general user-friendliness of *VirusBuster* over the last year.

One of the major nightmares of the last review was the log format – which is now much more standard in structure. In fact, the log files of the products on offer seemed in general to be approaching a common format which required very little tinkering to be parsed into final results. The only remaining major niggle is with those products which still list files in purely 8+3 format, *VirusBuster* and *Sophos Anti-Virus* being the primary offenders.

Returning to the review, *VirusBuster* receives a VB 100% award. It takes no great leap of logic to work out that full detection was achieved In the Wild, with no false positives.

CONCLUSION

The end of another *NetWare* review signals a traditional gap in comparative reviews, with the VB Conference at the end of September, and the Christmas holidays interfering with proceedings. As such, from a reviewer's point of view at least, it seems like the end of the year. Traditionally, I gnash my teeth in frustration at the state of *NetWare* products, though the situation seems a little more positive in this case.

Of the products reviewed last year only one has vanished. The others all look to be owned by stable companies who will not give up their support for *NetWare*. The buyout of *GeCAD* and partial burial of *RAV* can hardly be considered likely to be repeated with any other developer – even if a developer were the subject of a takeover, few potential purchasers have the financial freedom simply to ditch their purchases. To a *NetWare* administrator this will come as good news. Better news, of course, will be the fact that new products have been introduced for *NetWare 6*, albeit slowly.

So the range of products is there, but what about the quality? In this I must admit to have been pleasantly surprised. Issues which made life hellish last year have simply evaporated, replaced by features which are actually useful. Some oddities remain, but the feeling that the developers simply didn't care about users no longer prevails. Of those common problems which remain, setting on-demand scans is still difficult in many cases, so there is room for improvement. It will be interesting to see whether improvement over the coming year is as significant as that seen over the last.

On a final note, in last year's *NetWare* review I predicted that this would not be the year of the *NetWare* virus. This act of soothsaying proved successful, so this year I will go a step further. Not only will this not be the year of the *NetWare* virus, but by the time the next *NetWare* review is published, *Novell* will have released another massive service pack. Check back in 12 months and evaluate my psychic powers.

Technical Details

Test environment: Server: 1.6 GHz Intel Pentium 4 workstation with 512 MB RAM, 20 GB dual hard disks, DVD/CD-ROM and 3.5-inch floppy, running *NetWare 6 Service Pack 3*.

Workstation: 1.6 GHz Intel Pentium 4 workstation with 512 MB RAM, 20 GB dual hard disks, DVD/CD-ROM and 3.5-inch floppy, running *Windows NT 4 Service Pack 6*.

Network: 100 Mbit ethernet.

Virus test sets: Complete listings of the test sets used are at http://www.virusbtn.com/Comparatives/NetWare/2003/test_sets.html.

A complete description of the results calculation protocol is at http://www.virusbtn.com/Comparatives/Win95/199801/protocol.html.

NOTE CONCERNING THE LINUX COMPARATIVE REVIEW (*VB*, MAY 2003)

Concerns were expressed concerning some of the samples in the *Linux* test set following the results of the last *Linux* comparative (see *VB*, May 2003, p.18). These fell into two categories:

First, one of the samples of ELF/Siilov-5916 was found to be corrupt and non replicable. This has been removed from the test set.

Secondly, the samples in the *Linux* test set were copied from a *Linux* machine, to a *Windows* server, and then returned to the *Linux* test machine. During this process the *Linux* attributes – most importantly those denoting an executable file – were lost. It has been pointed out that these attributes are valuable in determining whether *Linux* files should be scanned, since extensions cannot be used for this purpose and may in fact be misleading. In future tests *Linux* executables and scripts will be marked with the correct attributes. In practice this should render one sample of ELF/Obsidian.E (with an extension of .EXT2) more easily recognisable as an object which should be scanned.

END NOTES & NEWS

The 10th International Computer Security Symposium, COSAC, takes place 14–18 September 2003 at the Killashee House Hotel near Dublin. A choice of more than 40 sessions and six full-day master classes is available. For full details of the agenda, venue, travel discounts, partner program and registration see http://www.cosac.net/.

COMDEX Canada 2003 will be held 16–18 September 2003 in Toronto, Canada. Discounted registration fees apply until 22 August. For details of the conference, exhibition and keynotes see http://www.comdex.com/.

The 13th *Virus Bulletin* International Conference and Exhibition (VB2003) takes place 25–26 September 2003 at the Fairmont Royal York hotel in Toronto, Canada. Full details, including conference programme, abstracts and information about the venue can be found on the *VB* website. Register online at http://www.virusbtn.com/conference/.

The 5th NTBugtraq Retreat takes place in the days immediately following the *Virus Bulletin* conference in Ontario, Canada. A welcome event on the evening of 26 September will be followed by the Retreat from 27–29 September 2003. Full details can be found at http://www.ntbugtraq.com/party.asp.

Black Hat Federal 2003 takes place 29 September to 2 October 2003 in Washington D.C. For more information and online registration see http://www.blackhat.com/.

InfowarCon 2003 takes place 30 September to 1 October 2003 in Washington D.C. Military leaders, representatives of political forces, academics and industry members will discuss the concepts of the latest on-going initiatives in the Homeland Security and Critical Infrastructure Protection communities. For details see http://www.infowarcon.com/.

The Fifth International Conference on Information and Communications Security (ICICS2003), is to be held 10–13 October 2003 in Huhehaote City, Inner-Mongolia, China. For full details see http://www.cstnet.net.cn/icics2003/.

The Workshop on Rapid Malcode (WORM) will be held 27 October 2003 in Washington D.C. The workshop aims to bring together ideas, understanding and experience relating to the worm problem from academia, industry and government. See http://pisa.ucsd.edu/worm03/.

COMPSEC 2003 will be held 30–31 October at the Queen Elizabeth II Conference Centre in Westminster, London, UK. This year's conference will include the Compsec 2003 Poster Session, featuring a review of the latest scientific advances in computer security research and development. For full details see http://www.compsec2003.com/.

The European RSA Conference will be held 3–6 November at the Amsterdam RAI International Exhibition and Congress Center, The Netherlands. For details of the agenda, location and registration see http://www.rsaconference2003.com/.

The Adaptive and Resilient Computing Security (ARCS) workshop will take place 5–6 November 2003 at the Santa Fe Institute, NM, USA. The aim of the workshop is to stimulate novel approaches to securing the information infrastructure. In particular the workshop will consider long-term developments and research issues relating to the defence of information networks. For full details see http://discuss.santafe.edu/bnadaptive/.

AVAR 2003 will be held on 6 and 7 November 2003 in Sydney, Australia. The theme for the conference is 'Malicious Code', incorporating emerging malicious code threats, the technologies at risk and the technology needed to deal with these threats both now and in the future. See http://www.aavar.org/.

COMDEX Fall 2003 takes place 15–20 November 2003 in Las Vegas, USA. See http://www.comdex.com/.

A selection of long-distance learning courses are to be run by the International Management Forum. Courses on Information Security, E-security and IT Service Management all commence January 2004. For more details see http://www. imf-online.com/.

SUBSCRIPTION RATES

Subscription price for 1 year (12 issues) including first-class/airmail delivery: £195 (US$310)

Editorial enquiries, subscription enquiries, orders and payments:

Virus Bulletin Ltd, The Pentagon, Abingdon Science Park, Abingdon, Oxfordshire OX14 3YP, England

Tel: +44 (0)1235 555139 Fax: +44 (0)1235 531889
Email: editorial@virusbtn.com www.virusbtn.com