OCTOBER 2003

virus BULLETIN

The International Publication on Computer Virus Prevention, Recognition and Removal

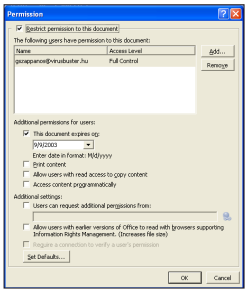
CONTENTS

IN THIS ISSUE

BIG, BAD AND UGLY

W32/Sobig is big, its code is bad and its style is ugly. In the absence of correct information, both speculation and wrong information have been plentiful. Peter Ferrie restricts himself to the facts about the six variants of Sobig, from A to F.

page 5



NOW YOU READ IT … NOW YOU DON'T

Office 2003's Information Rights Management provides methods for controlling which individuals can read and modify documents and can even set time limits on access to documents and messages. But Gabor Szappanos points out that this will have far-reaching consequences including headaches aplenty for AV support personnel.

page 14

ANTI-VIRUS WARS

We all know that it's not a good idea to attempt to install two or more anti-virus programs on the same machine at any one time, but Andreas Marx has discovered an interesting side effect of trying to use 'too much AV power': false positives.

page 17

ISSN 0956-9979

'Unfortunately, we sprinkle pixie dust on everything we do in an effort to market products.'

Russ Cooper
TruSecure Corporation

COMMUNICATING THE NAME OF THE GAME

Does anyone remember the name of the worm that appeared on 11 August 2003? W32.Lovsan.worm, Win32.Poza, Lovsan, W32.Blaster.Worm, Win32.Blaster, WORM_MSBLAST.A, Win32.MSBlast.A, Blaster, Worm/Lovsan.A, Worm.Win32.Blaster.6176, Worm.Win32.Lovesan, W32/Blaster, Win32/MSBlast.A, W32/Blaster-A, W32.Blaster, or Win32/Lovsan.A.

And that's just the initial worm. Then came the variants; Lovsan.B and W32.Blaster.B.Worm – but they aren't the same variant; they each have differently named executables. Then we get W32/Lovsan.worm.b, W32/Blaster-B, Lovsan.B, then W32.Blaster.C.Worm, Lovsan.C, etc. Need I go on?

I do not think it is possible for consumers to protect themselves solely through the use of software or hardware. They need to understand more about malware, about why it gets them, and what to look out for. In order for this to happen we have to create dialogue – not security mailing lists like *NTBugtraq*, but real dialogue where people talk face to face about these security events.

Unfortunately, we sprinkle pixie dust on everything we do in an effort to market products. If the consumer believes it's too complex for them to comprehend, and we reinforce that belief by constantly confusing them, maybe they'll buy annual update contracts and our next single-button solution to the problem. Some people in the industry seemed surprised that consumers were pummelled for two weeks in August 2003. After all, we *told* them to patch against the 'RPC/DCOM' vulnerability. We told them to apply 'MS03-026', which Windows Update calls '823980'. We increased ThreatCon, AlertCon, and InfoCon Security Levels. Clearly our current methods of communicating with the public are not working.

Several years ago I participated in an effort to create a single list of all vulnerabilities; the Common Vulnerability Enumeration project. The idea was simple: enumerate all unique vulnerabilities. One enumeration for a given vulnerability could be used across products which named things in different ways. Alas, there was resistance to the idea that enumerations wouldn't be consecutive. If we assigned a number to something, but then it turned out to be nothing – the number would go unused. So, instead of one number for each unique vulnerability, there had to be two. The first, the Candidate Number, is assigned when the issue is first raised. So you see vulnerability reports published with CAN-XXXX. When the candidate is finally accepted as unique, it is assigned another number, its final CVE-XXXX number. Of course, *SecurityFocus* adds a Bugtraq ID and other companies add their own ID numbers. In the end, how is the vulnerability best known? Code Red, Nimda, Slammer, Blaster, Nachi, etc. We can't even communicate amongst ourselves effectively.

I can't think of another profession which needs, so desperately, to communicate information effectively to the general public, but which cannot even discuss the topic within its own industry using common terms. We must figure out how we can work together to convey our message to the public.

We need some form of coordination between all parties involved in such events to ensure that the public receives reasonable information in an understandable format. Someone discovers a new worm, they assign it the next word in the dictionary, starting from A and working towards Z. That's the name that stays with it through its lifetime. *Microsoft* renames the patch to reflect the worm name and Windows Update reflects the change. No matter where you turn, any information about the worm contains the same name and same basic information.

I know this is all very cumbersome and we're all very busy with other things during security events. But just imagine what might happen if we made it simpler for the public to grasp these events – who knows, we might even get them to stop opening attachments!

NEWS

FOUR ARRESTS AND A CONGRESSIONAL HEARING

A US Congressional hearing was held last month to discuss the current state of Internet virus and worm attacks in the wake of the recent outbreaks of Blaster and Sobig.F. Members of law enforcement bodies and computer security experts including representatives from *Symantec*, *Network Associates*, *Microsoft*, *Cisco*, *VeriSign* and *Qualys* were called to testify before the technology subcommittee of the House Committee on Government Reform about issues surrounding the recent Blaster and Sobig.F attacks and how to protect the nation's computing systems from future virus and worm threats.

Some of the suggestions put forward for improving the security of the nation's computing systems included: better standards for producing secure software, computing ethics education aimed at children, increased funding and training for computer forensics teams and protocols for information sharing that would aid in capturing perpetrators across borders. Chairman of the subcommittee, Representative Adam Putnam floated the idea of legislation that would require publicly traded companies to complete a 'cybersecurity checklist' in their reports to the US Securities and Exchange Commission – potentially forcing companies to make changes in their IT security measures if investors or customers believe that an insufficient number of items are checked on the cybersecurity checklist.

The deputy assistant attorney general at the Criminal Division of the US Department of Justice, John Malcolm, faced some grilling when Putnam questioned whether cyber criminals face lighter penalties for the damage they create than other criminals – and whether, in fact, cyber criminals are pursued with the same vigour as other criminals. Malcolm argued that cyber criminals can be difficult to track and stressed that their crimes are taken seriously.

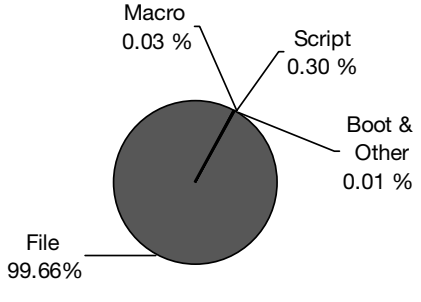
Meanwhile, in other corners of the globe, arrests of suspected virus writers were being made. Shortly after the arrest by US authorities of 18-year-old Jeffery Lee Parson on suspicion of creating and releasing W32/Blaster.B, Romanian police arrested a 24-year-old man suspected of releasing the .F variant of Blaster. If found guilty, it is reported that the young Romanian could face a maximum of 15 years imprisonment, thanks to the country's new computer crime laws. Less likely to face such a long sentence are two British men who were charged in connection with Troj/TKBot.A. The arrests followed an investigation by the National Hi-Tech Crime Unit which began in February this year. The pair have been charged with 'conspiring to effect unauthorised modifications to the content of computers with the intent to impair the operation of those computers' and await court appearance.

Prevalence Table – August 2003

Virus	Type	Incidents	Reports
Win32/Sobig	File	44,284	66.29%
Win32/Mimail	File	12,516	18.74%
Win32/Opaserv	File	3704	5.54%
Win32/Bugbear	File	1536	2.30%
Win32/Klez	File	1073	1.61%
Win32/Nachi	File	857	1.28%
Win32/Dupator	File	737	1.10%
Win32/Yaha	File	454	0.68%
Win32/Lovsan	File	342	0.51%
Win32/Funlove	File	245	0.37%
Win32/Fizzer	File	181	0.27%
Win95/Spaces	File	170	0.25%
Redlof	Script	113	0.17%
Win32/SirCam	File	68	0.10%
Win32/Ganda	File	59	0.09%
Win32/Magistr	File	59	0.09%
Win32/Gibe	File	55	0.08%
Fortnight	Script	54	0.08%
Win32/Kriz	File	42	0.06%
Win32/Hybris	File	34	0.05%
Win32/Holar	File	29	0.04%
Others		190	0.28%
Total		66,802	100%

[1]The Prevalence Table includes a total of 190 reports across 139 further viruses. Readers are reminded that a complete listing is posted at http://www.virusbtn.com/Prevalence/.

Distribution of virus types in reports



LETTERS

THE SUBTLE URGE TO COMMENT

Upon reading Juha Saarinen's Comment entitled 'The subtle return of MSAV' (see *VB*, August 2003 p.2), I was struck with the urge to comment on several of Juha's points.

My first point is that Juha recommends that we 'read up on' 'Windows File System Filter Manager Architecture' (WFSFMA) – which proves to be rather tricky. The only *Microsoft* documents that *Google* can find containing those words are either press releases or marketing documents – not to be trusted! It seems unfair to pass judgement without additional information – perhaps Juha has some that can be shared with *VB*? An article describing the technical details seems to be in order.

Next, Juha refers to the lack of an IPv6 firewall in *Windows XP*. Juha is unfortunate, in that even as the article was being prepared for publication, *Microsoft* released the 'Advanced Networking Pack' for *Windows XP* (which is available via Windows Update, and is described at <http://support.microsoft.com/?kbid=817778>). This contains the very firewall that Juha wished to see. But even discounting that point, I cannot imagine very many IPv6 users wanting to use a native (*Microsoft*-supplied) firewall.

For one thing, there aren't that many IPv6 users to start with, and most of them are IT professionals in large companies or universities who will need and have firewalls of their own, outside the client OS. In any case, the connection between the lack of such a client-side firewall and possible (unspecified and undescribed) flaws in WFSFMA appears non-existent.

In fact, Juha's point (with regard to WFSFMA) appears to be that *Microsoft* might get it wrong – that WFSFMA might not be perfect. This is a safe position to hold, but the article offers *Microsoft* no assistance. No suggestions are made in the piece as to how WFSFMA should be implemented, or what functionality it should provide and support.

Juha continues by mentioning that WFSFMA may 'thin the field' of anti-virus developers, and that such changes in the past have left vendors 'unable to distinguish their products from others'. However, it is not clear from the comments whether or not Juha considers this to be a bad thing. In the *NT* defragmentation space, for example, operating system API support led to increased product stability, reduced filesystem corruption, and was generally a good thing for users – is there reason to suspect that the addition of a competently-designed file system hooking API to *Windows* will be bad for users?

The comment finishes with the sentence '*Microsoft* would be wiser to put money into mending the broken security model in *Windows* that necessitates anti-virus solutions in the first place.' Whilst I am no apologist for *Microsoft*, I do think that such comments are unhelpful. Juha states that the *Windows* security model is 'broken', but does not discuss in what way it is broken, how it could be made better, what might be done to work around the problems, how the problems arose, or any such thing. This leaves the 'broken' claim as simply that – a claim. Again, an article describing the nature of the particular 'brokenness' to which Juha refers appears to be in order.

Ian Whalley

TIME TO CLARIFY?

I have a couple of comments about Richard Marko's excellent analysis of W97M/Lexar.A, 'Time to Relax?' (see *VB*, August 2003, p.10) – which are really only minor clarifications.

First, the article notes that the vulnerability that the virus uses (MS01-034) is still present in *Word 2000* and *XP*. It seems that the fix has now made it into the *Office* code, as the latest beta releases of *Office 2003* do not contain this bug.

Second, the article mentions that the offset that is modified by the virus points to a structure that contains data for the macros and toolbar customizations. This is correct, but I would like to add that, in this case, 'macros' refers to procedures that *Office* treats as macros, namely:

1. A procedure that is placed in a standard code module, and is called 'MAIN'.
2. A procedure that is placed in a standard code module, and is declared as Public.

In both cases the parameter list of the procedure should be empty.

These macros are the procedures that can be assigned to custom menu items or command buttons.

In a large number of macro viruses the virus code is placed in event handlers such as Document_Close, located in the default Thisdocument class module, and therefore not considered as a macro. Consequently, the corresponding macro description structure, as referred to in the article, is an empty structure, 0xFF 0x40, containing only the end byte.

But this does not mean that there is no virus or macro code in the document. It only means that there are no 'macros', as *Office* interprets the meaning of the term.

Gabor Szappanos
VirusBuster, Hungary

[*Gabor Szappanos expands on the subject covered in this letter in his article 'This message will self-destruct …', see p.14.*]

VIRUS ANALYSIS 1

SOBIG, SOBIGGER, SOBIGGEST

Peter Ferrie
Symantec Security Response, USA

W32/Sobig is big, its code is bad, and its style is ugly. In the absence of correct information, there has been speculation and wrong information in abundance. Let us restrict ourselves to the facts.

INITIALISATION

All known variants of Sobig begin by initialising their random number generator, using the current time as the seed. The first line of code, and here's the first bug already: the initialisation is carried out only once, in the main thread, but the random number generator supports multi-threading (using Thread Local Storage – see *VB*, June 2002, p.4), and the worm uses multiple threads. Thus, the random number generator is not initialised in those threads (the seed always begins at zero), resulting in the same sequence whenever the worm is executed. The same bug exists in some other multi-threaded viruses, such as W32/Welchia (described on p.10), and appears to be a common programming error.

Sobig.A checks the current date at this point. The second line of code, and – yes – here's the second bug. The worm converts the date to 'yyyy.mm.d' format, and compares the date against '2003.1.23' (23 January 2003). The problem is that specifying 'mm' requests the month in a two-digit form, which *Windows* dutifully supplies, using a leading zero for the months prior to October (the 10th month). This results in a date format of, for example, '2003.01.23' so the comparison always fails. The format should have been 'yyyy.m.d'. Later variants of Sobig check the date in a different way, which works correctly.

Sobig then checks whether it has been run with any command-line parameters. If it has been run without parameters, it assumes that it was launched by the user (either as an email attachment, or as a file that was copied across the network, for the variants that exhibit this behaviour).

When run without command-line parameters, the worm will compare its path name against the common path name that it uses on a compromised machine. If the two differ, the worm will attempt to make a copy of itself using the common path name. What might be considered a bug exists here too – the comparison of the name is case-sensitive, but *Windows* does not alter the case of a filename when copying over an existing file, so if the case of the worm filename is ever altered, the worm will attempt to copy itself every time it is executed without command-line parameters.

All known variants of Sobig attempt to copy themselves to the *Windows* directory (as specified by the %windir% environment variable), however the filename has been changed with each version. The list follows:

Sobig.A: winmgm32.exe	Sobig.D: cftrb32.exe
Sobig.B: msccn32.exe	Sobig.E: winssk32.exe
Sobig.C: mscvb32.exe	Sobig.F: winppr32.exe

If the copy fails (for example, if the user created a directory using the worm filename, as an attempt at a counter-measure), the worm will use the name of the currently executing file instead.

REGISTER NOW!

If the path names match, or after the copy is attempted, and, in the case of Sobig.D, no other copies of the variant seem to be running (another bug, see below), the worm will add itself to the registry, by altering the Software\Microsoft\Windows\CurrentVersion\Run key in both the 'LocalMachine' and 'CurrentUser' hives. This ensures that the worm will be executed whenever *Windows* is rebooted. The value has been changed with each version. The list follows:

Sobig.A: WindowsMGM	Sobig.D: SFtrb Service
Sobig.B: System Tray	Sobig.E: SSK Service
Sobig.C: System MScvb	Sobig.F: TrayX

All known variants of Sobig prior to Sobig.F have no command-line parameter in their registry data, so the initialisation code is executed whenever *Windows* reboots, adding to the overhead of the system. Sobig.F adds a command-line parameter ('/sinc') to its registry data.

After the registry has been altered, the worm executes itself again, this time with a command-line parameter, in order to proceed with the main execution. The contents of the parameter are never checked, only its presence or absence. Each version of the worm passes a specific parameter to itself, and the parameter has been changed with each version except Sobig.D and Sobig.E. The list follows:

Sobig.A: start	Sobig.D: dwaqr
Sobig.B: xcvfd	Sobig.E: dwaqr
Sobig.C: dwaqr	Sobig.F: /sinc

THE BIG EVENT

All known variants of Sobig use a named event in an attempt to prevent multiple instances of a variant from running at the same time. The name of the event has been changed with each version:

Sobig.A: Worm.X
Sobig.B: Mnkx.X
Sobig.C: Poss.X
Sobig.D: Nibs.X
Sobig.E: Nuiro.X
Sobig.F: TrayX

Unfortunately, the author(s) of the worm seem(s) to be incapable of mastering the required algorithm, despite several variations on the code in different variants. When a named event is created, the name is added to the global namespace of *Windows*, which means that the CreateEvent() API will not return a failure for an event that exists already (created by another process or even by a thread within the same process) – the same valid handle will be returned each time. The existence of the named event is that which should be checked. Instead, the worm checks if the event has been set (the worm sets the event whenever the initialisation code has completed). This results in a race condition that can, in turn, allow several copies of the worm to run at the same time – launched, for example, by someone clicking several times on the email attachment that 'doesn't seem to do anything'.

SOCKET 2.ME

After the worm's initialisation has completed, the worm will initialise Winsock support, requesting version 2.2, but ignoring the version that is returned, perhaps assuming that any version will be sufficient (which begs the question: why request such an advanced version?). Several threads are created at this point.

The thread details changed in Sobig.B and again in Sobig.F. Sobig.A creates a thread to notify someone (perhaps the author of the worm) by *ICQ* pager whenever the worm runs. That thread sends a mail from 'mail@mail.com' to '0@icq.pager.com', with a subject of 'Notify' and a body text of 'Hello'. This code is not present in Sobig.B and later variants, although the thread is still created in the variants prior to Sobig.F. Interestingly, the *ICQ* user name and body text were removed completely in Sobig.B, restored in Sobig.C–E (though the body text was changed to 'Worm started', and the '0' was changed to '1' in Sobig.E), and removed again in Sobig.F.

The remaining threads that are created are for the checking of updates, spreading by email, and spreading by network shares. In Sobig.F, the number of email threads was increased from one to seven.

BEST IF USED BEFORE …

Once the threads are created, Sobig.B and later variants will check the date on the local computer clock to see if their date of 'expiry' has been reached. The 'expiry' dates are as follows:

Sobig.B: 31 May 2003
Sobig.C: 8 June 2003
Sobig.D: 2 July 2003
Sobig.E: 14 July 2003
Sobig.F: 10 September 2003

If the expiration date has not been reached, the worm will enumerate the drive letters from A: to Z:, with a three-second delay between each drive, searching for non-removable drives. For each non-removable drive that is found, the worm will search recursively through all subdirectories, looking for files whose suffix matches one of those on the list that the worm carries.

All known variants of Sobig carry a list that contains: txt, eml, html, htm, dbx and wab. Additionally, the list in Sobig.F contains mht and hlp. The contents of files whose suffix matches one of those on the list will then be searched for texts that resemble email addresses. The worm uses OLE Automation to drive the VBScript regular expression engine to find these texts. The expression used by the worm is

```
[A-Za-z0-9]+[A-Za-z0-9_.-]+@(([A-Za-z0-9\-
])+[.])+[A-Za-z]+
```

which translates to: must contain at least one letter/number, followed by at least one letter/number/underscore/dot/hyphen, immediately preceding a '@', followed by at least one letter/number/hyphen and a dot (and this combination can appear multiple times), followed by at least one letter.

Since the VBScript regular expression engine returns the number of unique strings found, multiple addresses can be extracted from a single file. The worm checks its list before adding each address, and will not add duplicates. This list is used by the email thread(s). Additionally, Sobig.F keeps a list of the first 1000 filenames whose suffix is one of: jpeg, jpg, gif, htm, txt, doc, xls, mpg, eml, bmp, fax. This list is used by the network enumeration thread in Sobig.F.

Another bug exists here: when the drive searching routine completes, the worm will exit, regardless of what the other threads are doing.

UP, UP AND UPDATE

The update thread attempts to download a file from a server on the list that the worm carries. The thread contains no date check, so it continues to function even after the expiration date if, and only if, the drive searching routine is running to keep the worm active (as described above). Otherwise, no part of Sobig will function at all after the expiration date.

Sobig.A and Sobig.B will attempt to contact servers whenever the update thread begins to execute. Sobig.C and later variants synchronise themselves first with network time protocol (NTP) servers, and will attempt to download

files only during certain hours of certain days. The list of NTP servers to be contacted is carried by the worm:

129.132.2.21 (swisstime.ee.ethz.ch)

137.92.140.80 (chronos.ise.canberra.edu.au)

200.19.119.69 (server2.pop-df.rnp.br)

142.3.100.2 (clock.uregina.ca)

128.233.3.101 (non-existent)

193.5.216.14 (metasweb01.admin.ch)

131.188.3.220 (ntp0-rz.rrze.uni-erlangen.de)

131.188.3.222 (ntp2-rz.rrze.uni-erlangen.de)

212.242.86.186 (ogps.freebsd.dk)

chronos.cru.fr

138.96.64.10 (ntp-sop.inria.fr)

193.204.114.232 (unreachable)

133.100.11.8 (clock.tl.fukuoka-u.ac.jp)

193.67.79.202 (ntp0.nl.net)

193.79.237.14 (ntp1.nl.net)

132.181.12.13 (pukeko.cosc.canterbury.ac.nz)

150.254.183.15 (vega.cbk.poznan.pl)

62.119.40.98 (ntp1.sp.se)

200.68.60.246 (non-existent)

The server to contact is chosen randomly, however Sobig.E contains a bug that restricts the choice to only the first four entries. The list of days and hours is as follows:

Sobig.C: Monday, Thursday, and Saturday, from 8pm to 11pm UTC

Sobig.D: Thursday and Saturday, from 7pm to 12am UTC

Sobig.E: Monday and Friday, from 7pm to 12am UTC

Sobig.F: Sunday and Friday, from 7pm to 11pm UTC

Every two hours Sobig.A attempts to download the file from http://www.geocities.com/reteras/reteral.txt.

Sobig.B traverses its list by one row every two hours, and attempts to download from these sites:

http://www.geocities.com/fjgoplsnjs/jane.txt

http://www.geocities.com/lfhcpsnfs/mdero.txt

http://www.geocities.com/dnggobhytc/nbvhf.txt

http://www.geocities.com/bntdfkghvq/nbdcf.txt

Sobig.C traverses its list by one row every 59 minutes, and attempts to download from these sites:

http://www.geocities.com/vbifhdgs/aadfa.txt

http://www.geocities.com/vbhhrtok/axcfa.txt

http://www.geocities.com/vbhcbhptok/axccfa.txt

Sobig.D attempts approximately every 50 minutes to download from these sites:

63.187.136.207	65.92.185.105
218.145.251.172	4.46.216.107
63.139.177.178	68.158.97.35
68.119.94.107	65.96.174.173
211.172.37.81	80.133.8.182
203.218.1.205	65.96.134.32
80.193.162.47	68.160.246.76
217.86.31.254	68.51.149.158
24.159.40.38	24.101.46.49
24.199.119.153	67.85.144.168

Sobig.E attempts every hour to download from only the first five (because of a bug) of these sites:

67.164.250.26	80.145.119.84
129.244.36.194	61.41.223.43
67.73.60.121	218.158.43.206
218.146.139.246	67.168.13.135
66.169.84.77	209.34.8.147
64.229.253.52	65.69.221.166
65.95.29.173	67.74.161.243
203.252.75.45	80.136.150.140
217.230.224.66	69.22.34.186
65.95.91.31	62.47.6.238
217.228.235.145	24.96.26.108

Sobig.F attempts every hour to download from these sites:

68.50.208.96	65.92.80.218
12.232.104.221	63.250.82.87
218.147.164.29	65.92.186.145
24.33.66.38	65.95.193.138
12.158.102.205	65.93.81.59
24.197.143.132	65.177.240.194
24.206.75.137	66.131.207.81
24.202.91.43	67.9.241.67
24.210.182.156	68.38.159.161
61.38.187.59	67.73.21.6

Sobig.D and Sobig.E also listen on ports 995–999 for incoming data that can be used in addition to (or instead of)

the download server list. This is in the same format as the list which is downloaded from the servers.

Sobig.A–C will connect to the servers on port 80, using a standard *Windows* API to download the data; Sobig.D–F will connect to the servers on port 8998 to download the data. In all cases, the data expected to be received is the URL of a file to download and execute. Sobig.A uses no encryption at all; Sobig.B uses simple bit-twiddling of nybbles. Sobig.C–F use, according to Frédéric Perriot, a variant of DES which has the first and last steps removed. The lack of these steps does not seem to reduce the strength of the encryption. Additionally, the key generation algorithm was changed in Sobig.F, resulting in an incompatibility with previous variants.

Once the data has been downloaded, the worm will create or open a local download log file in the %windir% directory, then search the log file for a match of the data. Sobig.A attempts to match the entire URL, Sobig.B–F attempt to match only the filename part of the URL. If the data is not found in the log, it will be added to the log, after which the requested file will be downloaded and executed. The name of the download log file has been changed with each version. The list follows:

Sobig.A: dwn.dat	Sobig.D: dftrn32.dat
Sobig.B: msdbrr.ini	Sobig.E: msrrd32.dat
Sobig.C: msddr.dll	Sobig.F: winstf32.dll

EVER HEARD OF THE MAILMAN?

The email thread begins by creating or opening a local file in the %windir% directory that contains the 'sent' list. The name of the sent list file has been changed with each version. The list follows:

Sobig.A: smtmls.ini	Sobig.D: rssp32.dat
Sobig.B: hnks.ini	Sobig.E: msrrf.dat
Sobig.C: msddr.dat	Sobig.F: winstt32.dat

The worm will send one mail to each person who is not on the sent list. In the case of Sobig.F, however, the email threads are not synchronised, so it is highly likely that more than one thread (and potentially all seven of them) will send mail to each person not on the sent list. Additionally, if copies of the worm are running, all of those copies could potentially be sending mail to the same people.

For the sender's address, the worm will use its default address if only one email address is found or is remaining on the list; otherwise the choice will be made randomly from the list of email addresses, excluding the recipient's address. No attempt is made to discover the email address of the real sender, so if that address exists on the list, it could be used. The default address has been changed with each version:

Sobig.A: big@boss.com

Sobig.B: support@microsoft.com

Sobig.C: bill@microsoft.com

Sobig.D: admin@support.com

Sobig.E: support@yahoo.com

Sobig.F: admin@internet.com

The subject of the mail is chosen at random from a list that the worm carries. In the case of Sobig.E, a bug restricts the choice to only the first two entries. The list has been changed with each version. The list follows:

Sobig.A:

Re: Here is that sample	Re: Sample
Re: Document	Re: Movies

Sobig.B:

Re: My application	Approved (Ref: 38446-263)
Re: Movie	Re: My details
Cool screensaver	Your password
Screensaver	Your details

Re: Approved (Ref: 3394-65467)

Sobig.C:

Re: Application	Re: Screensaver
Re: Your application	Re: Movie
Re: 45443-343556	Approved
Re: Submited (004756-3463)	Re: Approved

Sobig.D:

Re: Application	Application Ref: 456003
Your Application	Re: Movies
Re: Accepted	Re: App. 00347545-002
Re: Screensaver	Re: Documents

Re: Your Application (Ref: 003844)

Sobig.E:

Re: Application	Application.pif
Re: Movie	Applications.pif
Re: Movies	movie.pif
Re: Submitted	Screensaver.scr
Re: Screensaver	submited.pif
Re: Documents	new document.pif
Re: Re: Application ref 003644	Re: document.pif

Re: Re: Document 004448554.pif
Your application Referer.pif

Sobig.F:

Re: That movie Re: Details
Re: Wicked screensaver Your details
Re: Your application Thank you!
Re: Approved Re: Thank you!
Re: Re: My details

The message body has been changed with each version. The list follows:

Sobig.A: Attached file: [attachment name]

Sobig.B: All information is in the attached file.

Sobig.C: Please see the attached file.

Sobig.D: See the attached file for details

Sobig.E: Please see the attached zip file for details.

Sobig.F chooses randomly from:

Please see the attached file for details.

See the attached file for details

The attachment name is chosen at random from a list carried by the worm. In the case of Sobig.E, the list is useless, since a bug restricts the choice to the first entry. The list has been changed with each version:

Sobig.A:

Sample.pif Document003.pif
Untitled1.pif Movie_0074.mpeg.pif

Sobig.B:

application.pif password.pif
movie28.pif approved.pif
screen_doc.pif ref-394755.pif
screen_temp.pif your_details.pif
doc_details.pif

Sobig.C:

document.pif 45443.pif
application.pif submited.pif **[sic]**
approved.pif movie.pif
documents.pif screensaver.scr

Sobig.D:

Application.pif ref_456.pif
Applications.pif movies.pif
Accepted.pif Document.pif
Screensaver.scr app003475.pif
Application844.pif

Sobig.E:

Your_details.zip (contains Details.pif)
Application.zip (contains Application.pif)
Document.zip (contains Document.pif)
Screensaver.zip (contains Sky.world.scr)
Movie.zip (contains Movie.pif)

Sobig.F:

movie0045.pif your_details.pif
wicked_scr.scr thank_you.pif
application.pif document_all.pif
document_9446.pif your_document.pif
details.pif

Sobig.E is the only known variant of Sobig that sends its attachments in Zip form. It carries the Zlib Deflate library in order to compress itself (as opposed to only storing, which requires no library code). Zip is a suffix that is not blocked by the *Outlook* security patch that prevents access to attachments based on their suffix.

The email part-separator is formed by appending a random eight-character hexadecimal value to a hard-coded text. For variants A to E the text is 'CSmtpMsgPart123X456_000_'. For Sobig.F the text is '_NextPart_000_'. The size of the attachment is variable in Sobig.E and Sobig.F. The file possesses a tail that contains the text 'XC001815d', prepended by a 32-bit value containing the number of bytes appended to the original file.

Before sending the attachment, the worm will add a random number of up to 3995 bytes to the original file, then include this tail. If the new size is larger than the old size then the tail text can be visible multiple times. Emails that are sent by Sobig.F include several 'X-' headers, one of which is 'X-MailScanner: Found to be clean'. Let us hope that no one is fooled by this.

When opening the file to attach, all known variants of Sobig do so in a mode without sharing enabled. This can result in a failure to open the file. The message is sent anyway, which is a reason why some emails that are sent by Sobig will arrive without the attachment.

SCHMOOZING AND NETWORKING

The thread for spreading by network shares is fairly standard. The routine is executed every four hours by Sobig.A, and every half hour by Sobig.B and later variants.

The worm enumerates the machines on the local network, and attempts to connect to the C$, D$, and E$, shares on each machine. For the variants of Sobig prior to Sobig.F, if the connection succeeds, the worm will copy itself to the root directory of the share, and to the Startup directories specific to US-English *Windows 2000/XP/2003* (Documents and Settings\All Users\Start Menu\Programs\Startup) and *Windows 9x/Me* (Windows\All Users\Start Menu\Programs\StartUp). Sobig.F begins by examining its list of the first 1000 filenames gathered by the drive-searching routine. If the list exists, a filename is chosen randomly from there, and '.exe' is appended to form a double extension (e.g. file.jpg.exe). If no filename exists, 'winppr32.exe' will be used instead. After the filename is chosen … nothing happens – the code to perform the file copy is not present in Sobig.F.

CONCLUSION

The storm of mail that was produced by Sobig.F might make it seem as though tens of millions of people were infected by the worm. This should give pause for thought, since multiple copies of the worm running at the same time, coupled with the multiple email threads, on a far smaller number of machines, could easily account for much of the contribution. Despite this, some variants of Sobig have spread quite well and very quickly. Sobig relies on minimal social engineering, and no exploits. Sobig.F does not even spread across the network, relying solely on email for its propagation. How has it become so successful? A recent comic explains (from *User Friendly* by J.D. 'Illiad' Frazer, (c) User Friendly Media Inc, userfriendly.org.):

Q. *After working for an ISP for over half a decade and being immersed in the web, what is the one lesson you've learned?!*

A: *Click on everything.*

Okay. Stop clicking. NOW.

W32/Sobig@mm	
Type:	Win32 SMTP mass-mailer worm, network share crawler.
Size:	65,538 bytes (A), 50,425 bytes (B), 57,241 bytes (C), 76,003 bytes (D), 86,528 bytes (E), 72,892 bytes (F).
Payload:	Can download and execute arbitrary executables without permission.
Removal:	Fix registry, delete worm copies and its data files.

VIRUS ANALYSIS 2

WORM WARS

Peter Ferrie, Frédéric Perriot and *Péter Ször*
Symantec Security Response, USA

Around 1966 Robert Morris Sr., the future NSA chief scientist, decided to create a new game environment with two of his friends, Victor Vyssotsky and Dennis Ritchie. They coded it for the PDP-1 at *Bell Labs*, and named their game 'Darwin'. Later 'Darwin' became 'Core War', a computer game played to this day by many programmers and mathematicians as well as hackers.

The object of the game is to kill your opponents' programs by overwriting them. The original game is played between two programs written in the Redcode language, a form of assembly language. The warrior programs run in the core of a virtual machine called MARS (Memory Array Redcode Simulator). The fight between the warrior programs was referred to as Core Wars.

Well, the world used to be a better place with the fights between genies in a bottle. Who let the worms out?

INSTALLATION

When Win32/Welchia first runs on a machine, it checks for the presence of a mutex called 'RpcPatch_Mutex', and aborts if the mutex already exists, in order to avoid running multiple instances of itself.

After creating its mutex, Welchia creates two services, one for the worm itself, configured to start automatically, and one for a TFTP server used during replication, configured to start manually. The service display names are 'WINS Client' and 'Network Connections Sharing', and the worm attempts to assign their descriptions from the legitimate 'Computer Browser' and 'Distributed Transaction Coordinator' services, respectively. The service executables are located in the %system%\wins directory and named DLLHOST.EXE and SVCHOST.EXE.

When started as a service, Welchia registers a basic service handler procedure with the Service Control Manager. (The handler procedure is kind enough to honour the STOP control requests, which makes it easy to stop the worm process on infected machines.)

BLASTER-BOMBER

Welchia attempts to remove Win32/Blaster.A from the machines it infects. More precisely, it kills any 'msblast' process by name (regardless of the extension, and the process name comparison is case-insensitive), and it deletes from the system directory any file named 'msblast.exe'

(after stripping from it a possible read-only attribute). This 'cleaning process' takes place immediately before the worm starts attacking new hosts, and periodically at the end of every infection cycle.

Welchia also checks for and attempts to install the MS03-026 patch for the RPC DCOM vulnerability on systems whose code page and locale match one of the following: China, Korea, Japan, Taiwan or US English. It duly reboots the machine after installing the patch.

By virtue of this curative effect, Welchia has been deemed by some to be a 'good worm'. This term is arguable for many reasons (incomplete coverage of the patching routine, unwanted side-effects, lack of control, etc.). The concept of a 'good worm' has been researched repeatedly, with little success. We would rather call it a 'jealous worm'.

COMPETITIVE ADVANTAGE

Welchia exploits two common vulnerabilities to infect new systems: the recent RPC DCOM vulnerability (MS03-026) already used by Win32/Blaster; and the NTDLL overflow from March 2003 (MS03-007) that the world came to know as the 'WebDAV' vulnerability, after one of the multiple paths that lead to its exploitation. (Indeed, Welchia uses the *IIS 5.0* WebDAV functionality to exploit the latter vulnerability.)

The use of multiple buffer overflow exploits to spread is new to Win32 worms (although it has been used in the *Linux* world by worms like Millen, Ramen or Adore). For Welchia, the use of multiple vectors is a competitive advantage in its fight against Blaster, because the pool of potential targets is increased compared to Blaster.

As a result of Welchia patching some machines against the RPC DCOM vulnerability, the digital environment becomes tougher for both Blaster and Welchia, but Welchia still enjoys a large base of unpatched IIS systems, giving it the edge in a 'survival of the fittest' race between the two worms.

ECOLOGICAL NICHE

After starting the TFTP server service that was created upon installation, and preparing the attack buffers for both of its exploits, Welchia starts its infection cycle, which has four phases targeting various ranges of IP addresses with different methods.

Before each attack phase, Welchia checks if its host is connected to the Internet by attempting to resolve the DNS name 'microsoft.com' to an IP address. If it can do so, it carries on with the attack, otherwise it waits 10 minutes and checks again.

The first attack phase targets the class-B network of the host with the RPC DCOM exploit. The class-B sized network is scanned linearly from top to bottom. Each IP address in this range is pinged. If the machine replies to the ping, the worm attempts to exploit it.

The second phase targets three class-B-sized networks (i.e. about 200,000 IP addresses) located nearby the host class-B with the RPC DCOM exploit.

The third phase targets one class-B-sized network, picked randomly from a hard-coded list of 76 target networks, with the WebDAV exploit. The target networks, most of which belong to Chinese organizations, were probably pre-scanned to find vulnerable machines. The WebDAV exploit carried by Welchia works only on some double-byte character platforms, which correlates with the geographic distribution of these targets.

Finally, in the fourth phase, the worm randomly selects either the RPC DCOM exploit or the WebDAV exploit and uses it against 65,536 random IP addresses. The addresses are selected from the following class-A network ids: 60 to 66, 128 to 172, 192 to 200, 202, 203, 210, 211, 218, 219, 220.

During all of these attacks, the worm avoids IP addresses containing the byte 0xc5, because it needs to patch the shell code with the bytes of the IP address xored with 0x99. Since 0xc5 xored with 0x99 would be 0x5c, which turns out to be the backslash character ('\'), the worm needs to avoid using it in both exploits, because they both involve over-long paths. As a consequence, the lucky few with IP addresses containing 197 (0xc5) will never be attacked!

SPL

When a machine is successfully exploited, be it with the RPC DCOM exploit or the WebDAV exploit, a connect-back shell code is executed on the victim. Unlike Blaster, which uses a binding shell code, Welchia expects the victim to open a connection to the attacking instance of the worm. The attacking instance binds a server that provides commands to the remote shell.

The server port was meant to be random in the range 666 to 765, but on most *Windows* systems the port ends up being 707 all the time. This is because the worm is calling srand() on the current tick count when the main thread starts, and then calls rand() from the server thread.

The random seed being a Thread Local Storage value on most versions of MSVCRT.DLL, the random number generator is practically uninitialized and always returns 41 on the first call (666 + 41 = 707). (For a description of Thread Local Storage, see *VB*, June 2002 p.4.)

Once a connection is established between the attacking instance and the remote shell, the shell command server on the attacking side issues some commands whose purpose is to make the victim download the worm through a TFTP transfer.

Unlike Blaster, Welchia does not implement its own TFTP server. Instead it carries around a TFTP server when it spreads. However, if Welchia comes to a new machine that has a file named tftpd.exe in the %system%\dllcache directory (this file is usually present on server flavours of *Windows 2000*), it will abandon the old TFTP server and snatch this new executable to carry around.

RPC VAMPIRE

The RPC DCOM exploit in Welchia is a stack buffer overflow very similar to that used in Blaster. The hard-coded return address to a 'call ebx' instruction used to hijack control is specific to *Windows XP* systems.

The exploit uses a connect-back shell code that opens a connection to the attacking machine, spawns a 'cmd' process whose input and output get associated with the socket connected to the attacker, and finally calls ExitThread().

The use of the ExitThread() API instead of the ExitProcess() API (used by the Blaster shell code) ensures that the RPC service survives the attack. Thus, machines infected by Welchia will not present the same symptoms as those machines infected by Blaster (unexpected rebooting, unavailability of some services, and so on …)

IMP SPIRAL

Welchia's WebDAV exploit hijacks an exception handler on the stack by overflowing a buffer in RtlDosPathNameToNtPathName_U (as do most published WebDAV exploits). From KiUserExceptionDispatcher, control goes to a 'call ebx' instruction in a 'well-known' data area of the inetinfo process (the same 'well-known' area to which we referred in our previous *VB* article on Blaster, see *VB*, September 2003, p.10). Details on the KiUserExceptionDispatcher function and how it relates to exception handler hijacking, can be found in Péter Ször and Bruce McCorkendale's article on CodeRed (see *VB*, September 2001, p.4).

Register ebx is then pointing to the hijacked exception record on the stack, one entry in a series of eight 8-byte exception records. Interpreted as CPU instructions, this series of exception records forms a ramp of pushes and pops designed to avoid execution of the exception handler addresses. Eventually, control flows to the beginning of a

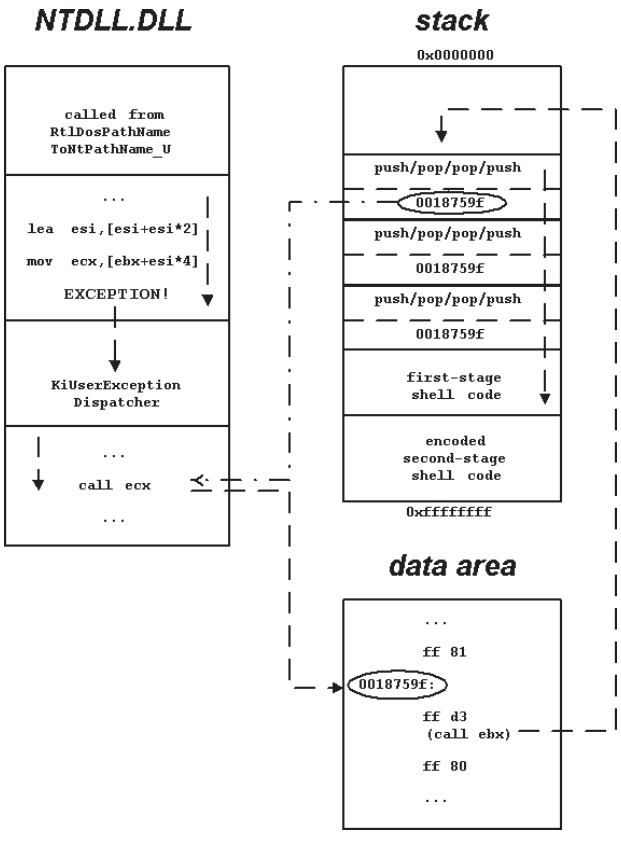


Figure 1: WebDAV exploit diagram.

first-stage shellcode. (See figure 1 for a diagram of the exploit phases.)

The exception record data are supplied as %u Unicode-encoded characters in the request URL. The Unicode-encoded characters are properly decoded only on double-byte character systems. In a typical US-English *Windows 2000* setup, question marks are substituted for the Unicode-encoded characters, and the attack results in a Denial of Service against IIS.

Interestingly, this WebDAV exploit uses a chain of three consecutive shell codes. It may seem suboptimal at first glance, because one shell code would be enough to achieve the same goal, but it results from a desire to separate the shell code functionality into reusable components. (Indeed, variants of the first-stage decoder, described shortly, were published previously.)

Once the control flow of IIS is hijacked, the first-stage shell code (supplied as Unicode-encoded characters) gets control, locates and decodes the second-stage shell code and jumps to it.

The second-stage shell code is encoded as a stream of lowercase letters that gets expanded by IIS, from ASCII to

Unicode, by inserting a zero byte in between consecutive letters. Each letter encodes four bits of information (a nibble). The first-stage shell code thus decodes one expanded dword into one byte of second-stage shell code.

The purpose of the second-stage shell code is to locate the third-stage and final shell code (shared between the RPC and WebDAV exploits). To achieve this, the second-stage shell code sets up an exception handler and strides through the process memory space, starting at 1 megabyte, in 16 kb increments (with 4 kb hops in case of page faults), looking for the ramp of Ns that constitutes most of the attack URL.

Once the ramp is found, the second-stage shell code scans forward byte-by-byte for the 'YXYX' marker that indicates the entry-point of the third-stage shell code, then jumps to this entry-point.

The third-stage shell code is the same as the one in the RPC DCOM exploit, described above.

POPULATION

As we write this article, Welchia attacks show no sign of slowing down. One of the authors of this article collected data on the ping probes coming from systems infected with Welchia (they can be distinguished from regular pings by their peculiar payload).

The data in figure 2 were collected from 24 August 2003, 17h00 (PST) to 3 September 2003, 23h59 (PST) and represent the number of hits from Welchia per one-hour time slice coming to one DSL IP address.

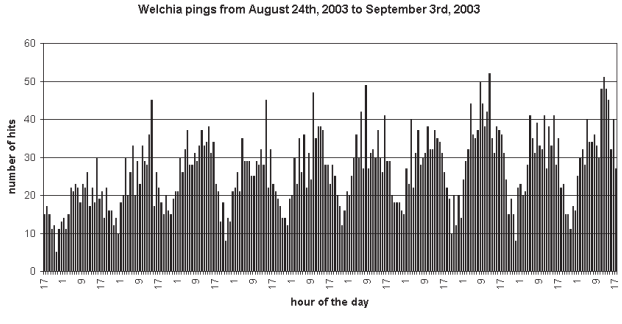


Figure 2: Welchia pings statistics.

LEMMING OF THE YEAR

Come 2004 and Welchia will jump off the cliff. Each time the worm starts it checks the current date, and if the year is 2004 it deletes its services and its files, and exits. It should be emphasized that the worm only checks the date when it starts, so that infected machines running unattended will not automatically stop attacking new systems on new year's eve. The service needs to be restarted, or the machine rebooted, for the wormicide to take place.

(As a side note, Peter Ferrie insists that lemmings do not really jump off cliffs to commit suicide. He claims they are in fact migrating to another island, and trying to get a head start because they don't swim very well …)

CONCLUSION

One genie jumped out of the bottle in about 1988, providing an early warning to all of us. Did we all learn the lessons of Morris worm? Sadly, it appears 15 years was enough to forget the warning.

Evidently worms will use multiple exploits in the future and will do so in more and more obfuscated ways. Under the attack of more than one major worm outbreak it is going to be increasingly difficult to provide accurate information about worm attacks.

Learn about the vulnerabilities in time, foresee all possible exploitation vectors ('WebDAV' was challenging in this respect), and recognize the exploit code in the blink of an eye. Without that, the information about protection might be only half-right or worse.

We have already highlighted the importance of worm blocking techniques. Not surprisingly, the basic principles to stop these attacks are the same for Morris, Blaster or Welchia (for both exploits) – however their description is beyond the scope of this article.

Will the entire Core Wars move to real networks in the form of new worm attacks? You could protect yourself, or play the ignorant for yet another 15 years. You decide!

Win32/Welchia

Size:	10,240 bytes.
Aliases:	W32.Welchia.Worm, W32/Nachi.worm, WORM_MSBLAST.D, Lovsan.D, W32/Nachi-A, Win32.Nachi.A, Worm.Win32.Welchia.
Type:	Exploits RPC DCOM vulnerability (MS03-026) and WebDAV vulnerability (MS03-007).
Payload:	Removes Win32/Blaster.A, patches some systems against MS03-026.

FEATURE 1

THIS MESSAGE WILL SELF-DESTRUCT...

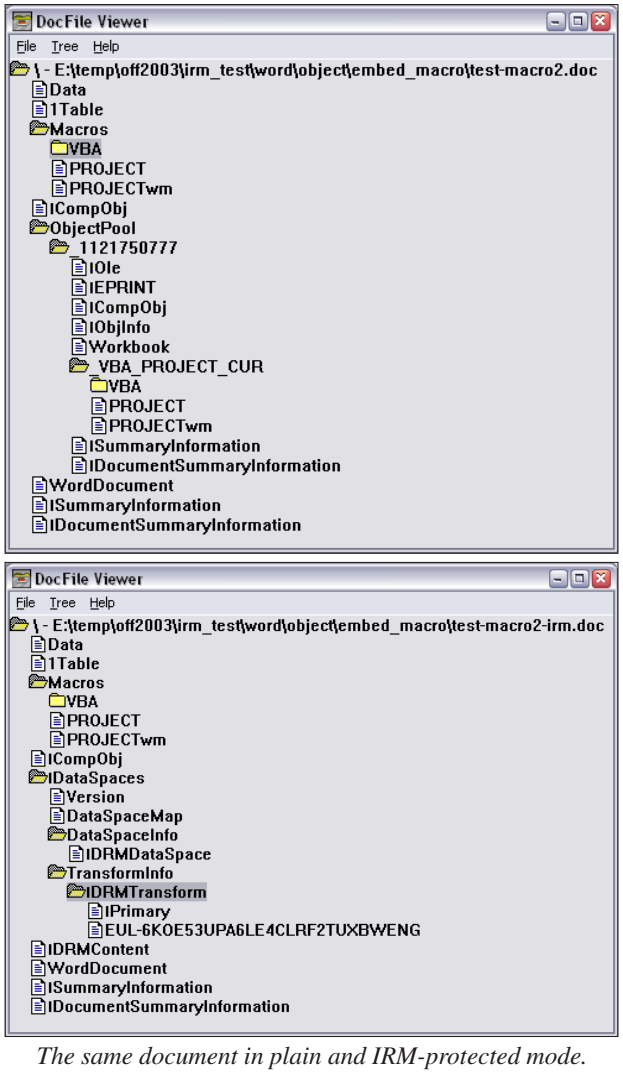
Gabor Szappanos
VirusBuster, Hungary

Office 2003 brings with it some new features, among them the improvement in document access management known as Information Rights Management (IRM).

This is the culmination of a long process starting from the password protection in *Word 6*, going through the Crypto API in *Office XP*, and finishing in *Office 2003*'s IRM. The purpose of IRM is to be able to control which individuals can read and modify a specific document. Crypto API did the same job, but the tedious work of key management was left to the end user. IRM removes this responsibility, assigning the key management to the Rights Management Server (RMS), usually set up in corporate networks.

So what does IRM provide? It is incorporated into the major *Office* applications (*Word*, *Excel*, *PowerPoint* and *Outlook*), and provides methods to control who can read and modify documents and messages, even giving time limits for access to documents. For example, it is possible to send an email message that can be read for a week; after a week the message expires.

This is achieved by encrypting the messages and the documents. Upon attempting to open a document the access right is authenticated against the user's rights and the validity of the certificate is checked with the RMS server. If the certificate is valid, the message is decrypted. If the user is not authenticated, or the message has expired, the content will not be available.



The same document in plain and IRM-protected mode.

OFFICE DOCUMENTS

Once again, *Microsoft* has 'enhanced' the file format of *Office* documents. For compatibility reasons, IRM-protected messages have dual format, so that both the old and the new *Office* versions can handle them – with different results, of course.

The main content of protected documents is transferred into encrypted streams. This includes embedded objects and pictures, but not the VBA project. The screenshots shown on this page illustrate the case of a *Word* document that contains macros and an embedded *Excel* worksheet (which contains macros itself).

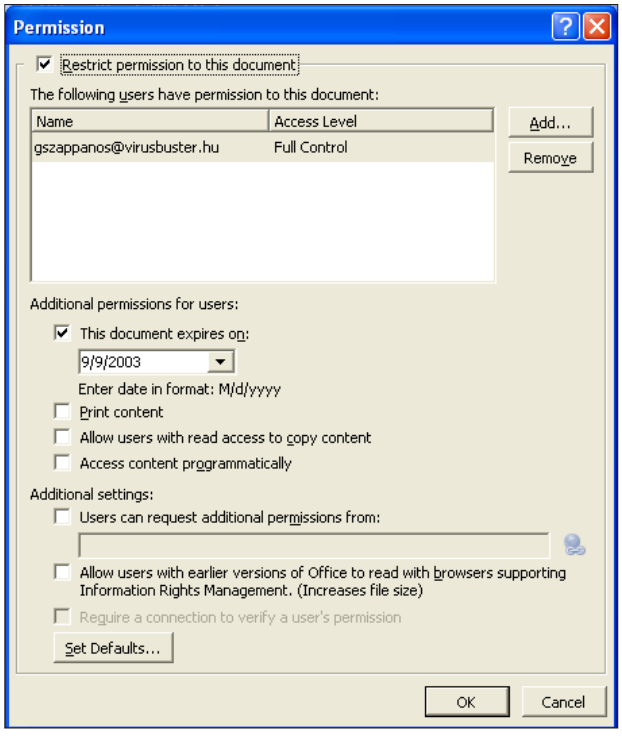
There is another thing that gets encrypted, and this is the *1Table* stream – which, among others, contains the macro table, with information about macros. The original stream is moved to the encrypted *Dataspaces* storage, while a new stream is created. When a document is decrypted, this stream is replaced with the original *1Table* stream on the fly.

This is going to have consequences.

I should point out that not every procedure gets registered in the data table, only the 'macros'. In this sense 'macros' refers to those procedures that *Office* treats as macros – namely procedures that are placed in a standard code module, that are not declared as Private and whose parameter lists are empty.

These macros are the procedures that can be assigned to custom menu items or command buttons, and can be executed from the *Tools|Macro* dialog.

In a large number of macro viruses the virus code is placed in class module event handlers such as Document_Close,



It is possible to send an email message that can be read for a week; after a week the message expires.

located in the default Thisdocument class module. Therefore the code is not considered as a macro, and consequently the corresponding macro data table is an empty structure, 0xFF 0x40, containing only the end byte. But this does not mean that there is no virus or macro code in the document. It only means that there are no 'macros', as *Office* interprets the meaning of this term.

If the document contains macros in this sense, some of the macro data (including the name, and key and menu item assignments) are stored in the document in the *1Table* stream. If a macro virus is disinfected by deleting its macros (as virus scanners usually do), the corresponding table items should be removed. If all macros are deleted, the table should be truncated to an empty table.

In any case, the table length data in the main WordDocument stream must be updated, otherwise *Word* will complain about the incorrect data structure, and the entire VBA storage will become inaccessible from within *Word*.

DETECTION PROBLEMS

Virus detection should not be a problem in IRM-protected documents, but there are still worrying scenarios in which current anti-virus scanners could fail:

- If the encrypted document contains an infected embedded document, the virus will not be accessible to the virus scanner.
- Even if a virus is detected, during the disinfection the scanner may need to modify data structures that are within the encrypted area. Error message boxes appear when an improperly disinfected document is opened – which will lead to a flood of support calls.
- Although they are not a significant threat nowadays, formula macro viruses reside in the workbook data area, therefore will be hidden from the scanners in an encrypted document.
- *PowerPoint* stores the macro storage as an embedded block within the main document stream. Using IRM the macros are not accessible to the virus scanners. While this is certainly a security hole, it is not a major threat, as there are no known *PowerPoint* macro viruses in the wild (except for the *PowerPoint* infections of the Tristate.C cross-application infector).

These problems are not at all new – they were present before, since earlier *Office* versions already encrypted the data streams. However, the encryption in earlier versions of *Office* was not as easy to use (partly because the key management for really strong encryption methods relied on the user); *Office 2003* will make it too easy to use, and users and companies will be tempted to do it.

Moreover, viruses using protected Crypto API are not likely to spread. This is because in order to infect other documents they have to manage the decryption-encryption process, which is quite complicated, so this is only applicable for virus droppers, but further generations pose no threat.

IRM, on the other hand, is transparent as far as the virus operation is concerned, therefore a virus can spread from protected document to protected document.

INFECTION ISSUES

The dual structure of IRM-protected documents raises several conversion and cross-infection problems. When infected IRM-protected documents are opened in an earlier *Office* version, the virus activates without problems, infecting the global template.

In the tests two viruses were used, Thus.FQ and Bablas.BS. The main difference between them is that Thus.FQ has an empty macro table; Bablas.BS works with macros, and has a full macro table.

There are two main scenarios of interest:

1. The document becomes infected and then IRM protected.

2. The IRM-protected document becomes infected under an older version of Office.

While the VBA storage itself is the same in both cases, the *1Table* stream is different. In the first case the table stream with the virus macros table is encrypted, and an empty table is present in the document. In the second case, the *1Table* contains the macro table with the virus macros.

What happens if infected IRM-protected documents are opened in earlier *Office* versions? In the case of Thus.FQ, nothing special happens, as both uninfected and infected documents have an empty macro table. Therefore IRM encoding makes no difference in this case; the virus will be activated from the document, and will infect the global template and further documents happily.

Bablas.BQ is a different case. In scenario 1 the macro table of the virus is encrypted and replaced with an empty table. When such a document is opened in an earlier *Office* version (*Office 97* and *2000* were used for testing, with identical results), *Word* will see an empty macro table, but as there is an ODE module in the document, it pops up the macro warning. In the VBE editor the macros are still accessible, but the automatic macros of the virus are not executed. For the operation of the automatic macros and the menu item hooks, *Word* gathers the data from the macro data table. Consequently, this sample will not work on older *Office* versions.

In scenario 2 when the IRM-protected document is opened in an earlier *Office* version, it will be locked for editing, thus the users will not be able to alter the content. But this will not stop the viruses, which infect the document and save it. This document save has a disastrous effect, as earlier *Office* versions will strip out all streams that are not recognized, including the encrypted main content of the document! Therefore, if an IRM-protected document is opened in a virus-infected older *Office* version, its entire protected content will be lost.

This presented a good opportunity to find out when *Word* will display the macro warning dialog. This will complete the pseudo-code presented by Richard Marko (see *VB*, August 2003, p.10).

The interesting fragment of his code is:

```
if (TestMacrosStorage())
{
        if (!WarnUserAboutMacros())
                return FALSE;
        UserWarned=TRUE;
}
CheckMacroDataTable();
```

According to Richard's analysis and my testing, the warning is always displayed if the macro data table contains data, independently of the content of the VBA storage – even if there is no VBA storage at all.

If the document has a VBA storage, but the macro data table is empty (there are no 'macros'), the warning is still displayed if there is any code module, class module or form in the VBA project (aside from the default Thisdocument class module) – even if these modules are completely empty.

Moreover, if there is only the default Thisdocument class module in the project, but it contains any code – even only an empty procedure – the warning is raised. So the functional pseudo-code is something like this:

```
BOOLEAN TestMacrosStorage()
{
if (num(VBA_project_items)>1) return TRUE;
if (len(code(Thisdocument))>0) return TRUE;
return FALSE;
}
```

DISINFECTION PROBLEMS

While in most cases the detection of viruses in the VBA storage does not pose a problem, the disinfection from encrypted documents may be problematic.

To carry out the disinfection properly, the virus scanner has to modify the macro table and its length record in the Worddocument stream (which is encrypted) – otherwise the *Office* applications will display error messages on opening the disinfected documents and the users will complain to their anti-virus vendors that their virus scanner damaged their document while disinfecting it.

Although the majority of the *Office 97+* macro viruses use class-module event handlers, which do not show this problem, some major families, like Bablas and all upconverted *Word 6* viruses are affected.

If the documents are infected with these types of macro viruses, it is impossible to disinfect them properly.

There is no problem with the disinfection of viruses like Thus, as the macro table is empty anyway, and the document will work after disinfection.

But in the case of Bablas there are serious issues. I will refer to the two cases as listed in the previous section.

In scenario 1 the disinfection clears the VBA storage. After that most virus scanners examine the macro table and wipe it out. As in this case it is an empty table, the scanner declares the document properly cleaned. Later, when opened in *Office 2003*, the *1Table* stream will be replaced on the fly with the stream containing the virus macro table. Then *Word* will assume it has macros (though the entire

VBA storage is missing) and (depending on the macro security settings) display the macro warning dialog.

Furthermore, *Word* will attempt to launch the automatic macros, as the macro data table has references to them. Not surprisingly, these are not found, causing another error message to be displayed about the missing macros or procedures. So, after a disinfection that claimed to be successful, users will get a virus warning and an error box. I guess our support folks will be kept busy explaining this.

In scenario 2 if the infection occurred in an older *Office* version, the IRM content is wiped out, the document is rendered to an ordinary document, and there will be no problem with disinfection – other than the loss of IRM content, which means another job for support.

OUTLOOK MESSAGES

Outlook also provides IRM functionality, similar to the *Office* documents: the recipients and the validity of the message can be controlled via IRM. In this case the attachments, the message body and the message HTML body are all encrypted and packed into a storage. The message can only be decrypted by the recipient's authorization.

Needless to say, this poses some security problems. It is easy to inject an encrypted backdoor or drop a virus inside an organization's defence lines, without a chance to scan it with an AV scanner at the entry point.

But there is no real virus spread, and it can be blocked at the client PC. However, there are several script viruses that attach themselves to outgoing email messages as an automatic signature, in the HTML body. These mainly target *Outlook Express*, but it is not impossible to do the same with *Outlook* as well.

In this case the infected user could send out infected messages, without the email virus protection detecting it.

CONCLUSIONS

All in all, IRM has not brought much to *Office* users that had not been seen before: users could use PGP encryption for protecting their email messages and Crypto API for protecting their documents.

All of these methods present possible security breaches for a company, but what IRM brings is the ease of use (no need for key management) and large numbers of possible users. This means that the existing, but so far mostly theoretical risks may become real-life experiences. The problems encountered with the disinfection of IRM-protected documents will cause support departments plenty of headaches.

FEATURE 2

ANTI-VIRUS VS ANTI-VIRUS: FALSE POSITIVES IN AV SOFTWARE

Andreas Marx, AV-Test.org
University of Magdeburg, Germany

I am sure that almost everyone working in the security business knows that it is not a good idea to install two (or more) anti-virus programs at the same time on the same computer – simply because each on-access guard wants to kill the other one … but that is not the only reason.

While performing a comparative review of anti-virus tools for a German magazine a few months ago, we discovered another interesting side effect of trying to use 'too much AV power' at the same time: false positives.

We found that *H+BEDV*'s *AntiVir* flagged the pavdll.dll file of *Panda Antivirus* as being infected by the W32/Kenston-1895.X virus.

Similarly, *Computer Associates*' *InoculateIT* (with the *CA* engine enabled) found Win32/Funlove.4099 in the file pavcl.exe (*Panda Antivirus* command-line scanner). Meanwhile, *DialogueScience*'s *Dr.Web* found Win32.Benny.6382 in the same file. And finally, *F-Secure*'s product identified a new variant of the Trivial virus inside one of the documentation files of *Kaspersky Anti-Virus*. What a mess!

THE REASON?

After a brief check of the files pavdll.dll and pavcl.exe an explanation for the false positives was identified: *Panda Software* does not fully encrypt its virus signatures and stores a lot of them in plain text, which is as they appear inside infected files. That was the reason why the signature scanning algorithms of *AntiVir*, *InoculateIT* and *Dr. Web* had flagged the files as infected.

Once we had identified the cause of the problem, we asked *Panda Software* if they would agree to fix it, by encrypting all of the virus signatures. However, the response from *Panda* was that, currently, this is not possible and that this is not their problem, because it is easy for an anti-virus program to see that these signatures are not a sign of an

infection. If other anti-virus companies would improve their tools to scan these files properly, the problem would not occur, they said.

We asked the other three anti-virus companies concerned for their opinions. Two of them told us that they could not do anything to avoid this problem and that the problem could only be fixed by *Panda Software* encrypting their signatures.

CA was the only company to try to fix the problem by altering its scanning engine. This was because *CA* had also received a number of customer complaints about false positives in the – guess what – only partly encrypted signature file pav.sig.

For a limited time, the string 'Signature file system (c) Panda Software' (the header of the pav.sig file) was visible in the *CA* engine (avh32dll.dll) to ensure that the scanner would skip this file during a scan and avoid generating this false positive.

The false positive generated by *F-Secure Anti-Virus* (which has two main scan engines, *F-Prot* and *Kaspersky*) was caused by the presence of the eicar.com file inside a short part of the *Kaspersky Anti-Virus* documentation. The *F-Prot* engine found this suspicious.

OTHER EXAMPLES

The examples described above are the result of just one test! In the past we have encountered several other problems like this, and in the majority of cases they were caused by plain, unencrypted signatures.

Examples include a routine in *AntiVir* which was written to clean systems infected with Win32/Qaz. In order to restore the registry, *AntiVir* stored the strings 'StartIE' and 'qazwsx.hsq' in plain text to delete keys created by this worm. This was enough for *Network Associates*' *VirusScan* to flag the anti-virus tool as being a possible new variant of the Win32/Qaz worm.

In this case, however, both companies fixed the problem in their next product releases: *AntiVir* encrypted the text strings and *Network Associates* extended the driver to check for more than just this signature in order to report a new variant of an existing virus.

APPORTIONING THE BLAME

But not every anti-virus company fixes problems like this silently, as illustrated by the following text which was linked from the *F-Prot* website for quite some time but has since been removed (source: http://www.f-prot.com/f-prot/news/noworm.html):

'**The RealTime Protector component is not a worm**

Mcafee's antivirus product, using definition files number 4199 , falsely detects the RealTime Protector component of F-Prot Antivirus as a new worm. This problem with the Mcafee product applies to machines running Windows NT, 2000, and XP with F-Prot Antivirus 3.12.

Needless to say the RealTime Protector component of F-Prot Antivirus is not a worm, neither a new nor an old one. The source of this problem lies solely with Mcafee's apparent lack of quality control.

Mcafee users encountering this false detection of the RealTime Protector component are encouraged to ignore it and upgrade their definition files when newer files become available from Mcafee Inc. when they have fixed this problem. Users of Mcafee can also upgrade to F-Prot Antivirus for Windows here to get a more secure and reliable antivirus protection.'

As I was writing this article, I received a question about *Kaspersky Labs*' clrav.com utility, which cleans PCs infected by worms such as Win32/Opaserv. The download of the utility was blocked by *NAI VirusScan*. The heuristic reported the following: '*Found virus or variant New Worm !!! Please send a copy of the file to Network Associates*'. So I did.

CONCLUSION

Anti-virus tools from one company often have problems co-existing with the tools from another, especially in the area of false positives. Some of these problems could easily be avoided – the developers would only need to store their virus signatures properly encrypted in all parts of the program, the engine and the virus definition files. Not only should the signatures be encrypted to avoid false positives, but also to provide a form of protection against virus writers (who, having access to the easily-visible signatures can create new variants using different patterns) as well as protecting the company's intellectual property.

A simple runtime-compression or encryption of the whole executable file is not a viable option, because many anti-virus tools are able to uncompress or decrypt such programs easily. Therefore they would still find the signatures that caused the false positive.

In addition, the detection routines of a number of anti-virus programs should be fine-tuned so that a single short signature found in a file does not result in a virus alert at all. Last but not least, it is important for anti-virus vendors to have a copy of all competitors' programs (including the most recent updates and special cleaning tools) in a false positive test set which should be scanned before releasing a new definition update.

FEATURE 3

A HOP TO THE PIRATE SHOP

Jong Purisima
TrendLabs, Trend Micro Inc., Philippines

Amazed and inspired by Eddy Willems' tales of his adventures in Saudi Arabia in his article 'Virus Hunting in Saudi Arabia' (see *VB*, August 2002, p.12), I decided to embark on a similar quest. Living in a region where software piracy is rampant, I was interested in checking out the local scene.

My goals were simple: to find out whether viruses are being sold in pirated collections and to check on the availability of pirated anti-virus software. I was also curious to find out whether the people selling pirated CDs (or DVDs) really understand what viruses are – and if they do, whether they are aware of the contents of what they are selling.

My strategy was to pose as a student gathering data on whether viruses lurk in the software collections sold. I would approach merchants and ask if I could borrow a couple of CDs to check for virus infection.

SETTING THE SCENE

Greenhills is an area in San Juan situated at the heart of Metro Manila. It boasts a cluster of malls that cater to almost everyone's every need. Having established itself as a shopping destination in the mid-1970s, in its first two decades, Greenhills was known as the place to buy imported goods. Lately, however, it has become popular for imitation merchandise – with prices pegged, on average, at 5%–25% of the original retail price.

Greenhills offers a wide array of merchandise – numerous stalls peddle clothes, shoes, bags, jewellery, and household items. Mobile phone trading is also a booming business. Pirated VCDs and DVDs are one of the main attractions that draw people to Greenhills. Shelling out $1 for a VCD and less than $2 for a DVD is not a bad deal – while a VCD player is usually priced at $30 and DVD players range from $40.

Greenhills is also the spot where computers and accessories can be bought at low prices. The cluster of stores creates heavy competition amongst outlets for both prices and services. These merchants do not sell computers with pirated software installed, but if you insist, they will advise you to buy the software from the pirate stores around the area and they will assist you with installing it.

THE CHERRY

My target merchants were those selling pirated software – ranging from operating systems, applications and development tools, to MP3 collections, picture compilations, digital encyclopedias, games, tutorials, digital references and all the other things that are offered as software nowadays. These stores are not like the 'one-stop shop' that Eddy Willems came across in Saudi Arabia, since most of the disks are already compiled and are sold along with the fancy CD covers that go with them. What I was hoping to get hold of were those disks that contained viruses in software compilations.

THE FIRST HOP

Upon arrival at Greenhills I went directly to the spot where software collections are sold. At the first software store I explained that I was a university student writing my thesis paper about viruses in software collections. I told the merchant that I would like to help him in checking whether there were any viruses in the CDs he sold. At first, he misunderstood what I was asking for and offered me anti-virus software 'Ah yes the latest Norton and McAfee scanners,' he said, 'they just arrived from Malaysia two nights ago. We also have PC-CILLIN 2003.'

Since this was not what I wanted, I explained my offer further and told the man that I would take home the CDs and scan them using multiple virus scanners and return them with the feedback as to whether they contained viruses or not. However, the merchant was not convinced that I was a student and told me that he could not help me further.

I received the same response from the next 10 merchants or so – they were not willing to take a risk. After a number of exhausting exchanges, I concluded that anti-virus products from all the major vendors are very much available here. Not only that, the stores also offer anti-virus and security software for every tier of protection – the gateway, email servers, retail desktop, corporate-wide and even have products compiled in suites.

A notable find was a CD named 'Antivirus Pro 2003' which contained 34 anti-virus products and some tools like PGP and Black ICE divided among some 15 vendors. The collection contained almost every new version of the latest anti-virus products for the desktop market from a number of anti-virus vendors. (I was particularly surprised to find a version of *Virus Striker* which, if I remember correctly, is

the US version of the *VirusBuster* AV software produced in Hungary.) Most of them came with a 'SERIAL NUMBER.TXT' file. One had a serial number key generator together with the installation program.

I finally stumbled upon my target in the 12th store. The merchant handed me a CD entitled 'Hackers World 2003', with a list that included an entry entitled 'Virus Coll 2003 (Be Careful)'. I asked her if she had any more CDs with similar content and whether she would be willing to take part in my research. She brought out 20 more CDs purporting to be hacker tools and tutorials. Browsing through them, I found out that only a few contained the word 'virus'.

I asked her if I could take a few home to check out for infection. She agreed, I promised to return the CDs in a matter of days and also took home a few 'best-sellers' – mostly OS installations and applications.

Once I got back to my test machine, I scanned the CDs using nine AV products. The results of the scanning of the Hacker Tools and Tutorials CDs are shown below:

CD name	No. of files scanned	No. of infected files	Total unique malware
'Hacker's World 2003'	28,644	552	320
'Hackers XP 2002'	23,446	390	259
'The Hackers'	65,146	409	285
'Hackers 2003'	69,720	75	53
'Hackers Encyclopedia 2002'	60,056	9	5

The malware found on the CDs comprised Trojans, backdoors, nukers, spyware, a few virus kits and a few old DOS viruses that came paired with their respective ASM source code. These were linked to the tutorial texts which teach the user how to hack using the tools provided on the CD and a short discussion on how viruses work and how to write them.

The operating systems CDs were clean but some of the files on the application compilation CDs were flagged as CIH damaged or corrupted. Upon further analysis, these proved to be cleaned files that had previously been infected by CIH. This was also the case for some of the executables on the hackers' CDs.

THE SUCCEEDING HOPS

As promised, I returned the CDs to the merchant a couple of days later. In the meantime she had explained my quest to a number of other traders, a few of whom offered to give me some CDs that had been returned to them by customers. I took home five suspected CDs plus five new best-sellers.

Upon scanning the 10 CDs, the five new best-sellers came out clean, while the other five logged a number of damaged or corrupted files as before – but this time not only with CIH but with FunLove as well.

I returned to the shopping mall and spent a long time explaining to the traders why the corrupted files had been detected and why it was still somewhat safe to execute them. However, the traders were angry that this had happened and upset that they had accepted imperfect goods. I thanked them for their help and went on my way.

I returned to the pirate shops two months later to check on how things were going. This time the merchants gave me piles of CDs to scan. I asked them why they didn't simply use the anti-virus products they were selling for this purpose. They laughed and told me that they did not think they would be able to interpret the results by themselves. I took home 10 more suspected CDs and 10 more best-sellers. Once again, each of the suspected CDs proved to contain at least one file infected with a damaged or corrupted CIH or FunLove, while the best-sellers were clean.

THE LAST HOP

A few days before finishing this article, and around four months after my initial visit, I returned to the pirate shops one last time. I browsed through their latest anti-virus collections to find out if any of the recent soon-to-be-released AV products had already made their way into the pirates' hands. I found none, but I believe they will be available here before the end of the year. I asked whether customers still returned CDs due to infection and was told that none had been returned since my last visit.

RECAPPING THE EXPEDITION

I was told that, following my second visit, the traders expressed concern about the infected CDs to their supplier in Singapore. Since then the supplier has been using two or more anti-virus products to scan the compilations before starting the mass reproduction. The traders don't want their collections to contain viruses since it's bad for business. They try to delete every virus they encounter, with the exception of those on the hacking tutorial CDs.

On reflection, I was pleased to find out that even though people in this corner of the world do not know much about malware and exactly how it works, they plainly understand that it is a nuisance: they don't want it and they want to get rid of it. They may be trading unethically, but they too are affected by the nasty bite that malware brings to this world – even those in the not-so-legal line of business try to do the right thing in ridding the world of malware.

PRODUCT REVIEW

NOD32 ANTIVIRUS 2.000.6

Matt Ham

Despite being a long-running entrant in the *Virus Bulletin* comparative reviews, there are rarely many words to accompany the basic details of *NOD32*'s test results. This is due to the lack of either major changes to the product or any outstanding flaws – something which might be considered a bit of a disadvantage to the author of a standalone product review. Recently, however, *NOD32* has undergone a cosmetic transformation and a number of new features have been added to the product.

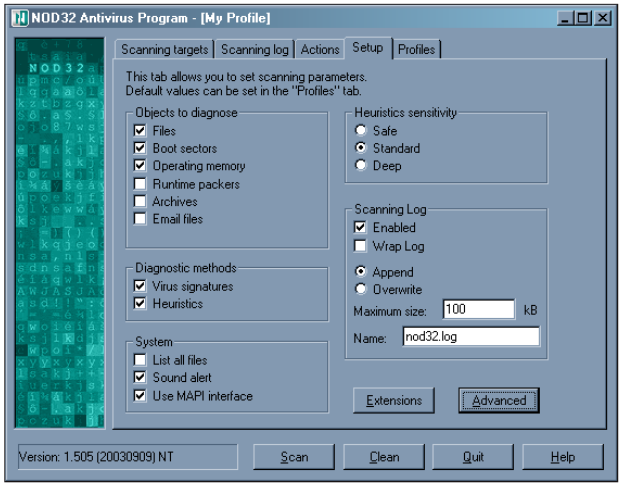
The most notable addition to the product over recent years is that administrative support is provided now. Of the negative comments levelled at *Eset*'s product, the vast majority have been based around its lack of scalability. The administrative options could thus have a large effect upon the perception of *NOD32* and will be examined in most detail.

INSTALLATION

The version supplied for testing was the Administrator version, a title which might suggest lengthy and complex installation routines. Happily, this was far from the case.

The familiar InstallShield three-fold offer of configurations is the start point in the installation procedure. In this case Typical, Advanced and Expert installs are the choices – as might be expected, these offer increasing levels of interaction. The Expert installation was used for testing. If a reinstall is performed, there is the useful option to retain all previously selected settings.

The first choice to be made is whether silent mode should be activated, in which case 'Important messages' will be sent to the administrator, rather than to the user. Silent mode does not, however, do away with the need for user interaction – making this seem rather an odd selection. Setup parameters may also be password-protected, and it is apparent that the combination of the two options will allow for what is more usually described as a silent install. Inbuilt installation parameters which override user choice and offer no interaction are a requirement for cloning installations to user machines – therefore the enhanced administration features are certainly no myth.



The next choices to be made are whether the *Eset* splash screen should pop up at system startup, and whether the *Eset* or *Windows* 'look' should be used for the GUI. Screenshots in this review are in the *Eset* style, although the *Windows*-style version was not noticeably different to my (admittedly non-artistic) eye.

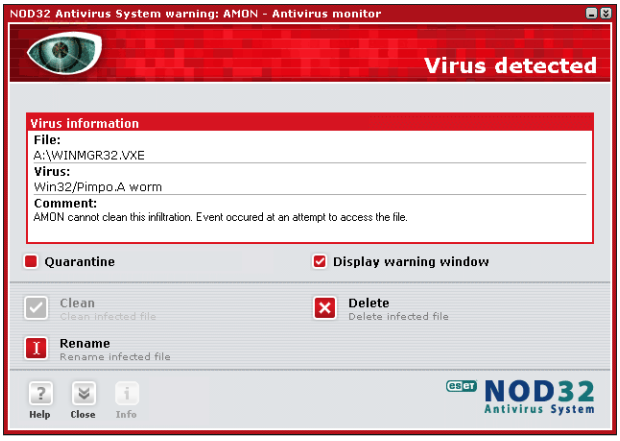
Warning messages are next to be considered, with options as to whether email or *Windows Messenger* should be used for sending alert messages. Either or both may be chosen, although the default setting is neither.

The next option is more interesting in that it defines parameters for the automatic update of the software. In this case only Internet updates are considered. It is impossible to disable the updates without having made a conscious decision to do so, which is reassuring. With this in mind, it is a little odd that the on-access scanner is recommended for automatic loading at start-up – but this is not the default option.

More mundane matters are chosen next: whether right-click scanning is to be enabled and whether a desktop icon should be added for the on-demand scanner.

The email scanning features of *NOD32* are provided by an application named IMON, which is activated by default. Due to the wide variety of email clients available today, the developers have conceded that not all will be fully compatible. Users of unusual email clients may thus select the IMON setting which offers greater universal compatibility at the expense of some features or of speed of scanning. The IMON setting is selected using a slider bar, ranging from maximum efficiency at one end of the scale to maximum compatibility at the other. The default setting here is for maximum efficiency. By default a notification is added to scanned emails; this can be eliminated or added only to those which are infected.

At this point the installation is ready to complete. Once the file transfers are finished the system must be rebooted – which could be an irritation when installing to servers. The installation process results in a program group containing the Configuration Editor, NOD32 Control Center and *NOD32* itself. Help, readme and uninstallation files are also located here.



DOCUMENTATION AND WEB PRESENCE

Documentation is provided in the form of a PDF manual and built-in help files. The manual covers only the standard installation process – the other two types of installation are mentioned, but no more than a small chart is devoted to them. Since there is no help function available during installation, the lack of information in the manual was disappointing.

When using the application itself, the manual tended to take second place to the built-in help. Since the help feature is context-sensitive, information about features that are in use can be obtained pretty much instantly – enough of an advantage to render the manual nearly obsolete from the point of view of a feature reference.

Clearly, the authors of the documentation have realised this, and it is reflected in the content of the manual. Rather than providing lists of features the manual is more concerned with such activities as post-installation checking of scanner operation, what steps to take if a virus is detected and troubleshooting.

The *Eset* website can be found at www.nod32.com and holds no great surprises – virus information, product news, support, download areas and everything you would expect from an anti-virus vendor's website.

THE SCANNER

NOD32 contains all the standard features of an anti-virus application, in this review only the more unusual or notable will be mentioned. The Setup and Profiles portions of the application are those which call for close inspection.

Of note, due to the comments made in the help file, is the choice of which objects should be scanned. This defaults to Files, Boot Sectors and Operating memory. By default, runtime packers, archives and email files are not scanned on demand. It is recommended in the help files that archive scanning be disabled by default. Although it may be argued that scanning of archives can be time-consuming, on-demand scans would seem an ideal opportunity to do this. Since the on-access scanner defaults to requiring a manual load, the scanning of archives would seem more important than if archives were guaranteed to be scanned on access when opened.

It is also notable that three levels of heuristics are available: Safe, Standard and Deep. Heuristics may be disabled totally. This is fairly standard, although in this case it is also possible to disable virus signature scanning. Thankfully, it is not possible to disable the two scanning methods simultaneously.

The profiles section is less contentious, allowing configurations to be saved for later use. These options may be password-protected, enabling a default profile to be enforced upon end users, while giving administrators more leeway.

CONTROL CENTER

The Control Center is the primary administrative tool for *NOD32*, though it is likely to be used alongside the Configuration Editor which is discussed later.

The initial interface for the Control Center is positively tiny – a tree of objects and three buttons for help, minimising and leaving the application. The tree has four branches: Resident modules and filters, Update, Logs and NOD32 System Tools. When first launched there is no right-hand pane – a docked right-hand window opens when a branch or sub-branch is selected.

The Resident modules and filters branch allows access to the local versions of AMON (the on-access scanner), IMON (the email scanner) and the *NOD32* scanner. It is possible to set parameters for the AMON and IMON modules directly here for use on the local machine – parameters can also be set indirectly for *NOD32* by invoking that program from a shortcut. Shortcuts are also provided for scanning local drives and disks, and statistics and version numbers are available.

The Update branch allows the same sort of access to and control of the Update process. It also contains a sub-branch entitled Mirror. This is used in cloning update files that are available from other servers, so that they may be used by other machines. With this functionality it is relatively easy to envisage a single administrator's server creating a mirror of the central *Eset* server, with further internal servers pulling updates from here. This allows for central updating of machines within networks and is a crucial addition for users within all but the smallest of networks.

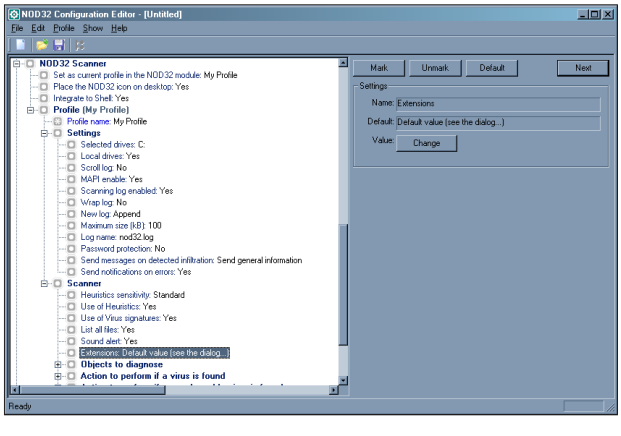
The Log branch is self-explanatory, having areas for viewing event logs, virus logs and the *NOD32* logs. Virus logs are produced by the components other than the on-demand scanner. Although the logs may be inspected and are easily available in this central point, there is no facility to sort, filter or otherwise manipulate them.

The final branch, NOD32 System Tools, is home to the Quarantine, Schedule Planner, Information and System Setup sub-branches. Quarantining and Scheduling offer no surprises. The Information page summarises information about versions for the various components of *NOD32* and gives some system information. There is an option to dump the information to the clipboard – this is clearly a feature designed for ease of technical support or troubleshooting. System Setup is a reiteration of the choices made at installation.

CONFIGURATION EDITOR

As with most anti-virus applications, the *NOD32* program is replete with configuration options, which may be selected, changed, altered and generally messed about with to the heart's content. The interface of an anti-virus program is, however, designed more with scanning in mind than for ease of configuration. It may be necessary – especially as an administrator – to alter or completely substitute configuration settings on multiple machines. The Configuration Editor is provided for exactly this reason.

Configuration files are stored in XML format and include within them full details of all *NOD32* profiles defined for the machine upon which they reside. Storing these profiles within the configuration file ensures that when configuration files are distributed, rogue profiles will not exist containing contradicting policies. Furthermore, a large amount of additional configuration information is provided – which goes well beyond the options required by a general user.



The areas which are controlled within the configuration files cover all those which are available from the setup dialogs as well as IMON and AMON (the email and on-access scanners), updates and mirror settings. 'EMON' settings are also included – though it is not immediately apparent what facet of the product is controlled here.

With an XML structure, the configuration files are easy to edit and inspect using a range of editors. This is slightly worrying, since configuration files can hold numerous passwords and user names, depending upon which accounts they must access in the course of operation and update. However, a brief inspection of the internal data shows the password details to be encoded in a non-obvious fashion. While this cannot be considered to be completely secure (since it is relatively easy to gain plain text-encoded pairs by the insertion of arbitrary password strings into the *NOD32* client), the matter has at least been given some consideration.

CONCLUSIONS

NOD32 is not alone in having a large group of dedicated followers who use the product on home machines or small networks – but has fewer fans among administrators of larger networks. The administrative features that have been added since the product's last review might go some way towards changing this situation. True, there are further administrative features which could be added, but the existence of an integrated framework will enable these to be added gradually.

The primary administrative feature still lacking – and rather a major one – is remote installation of clients. Although there are several features and options available which would appear ideal for producing an executable package so as to install a pre-configured version of the software automatically and silently, this is not something which is mentioned in the documentation. It would not be hard to create such an executable and install it using a login script – though there is certainly room for more user-friendliness here. It is clear that new features are still being added, however, and this feature is under development.

Technical Details

Test environment: Identical 1.6 GHz Intel Pentium machines with 512 MB RAM, 20 GB dual hard disks, DVD/CD-ROM and 3.5-inch floppy drives. The primary test machine was running *Windows 2000 Advanced Server SP2* with *Windows XP Professional Clients*.

Developer: *Eset s.r.o.*, Svoradova 1, 81103 Bratislava, Slovakia. Tel +1 619 4377037; fax +1 619 4377045 (US corporate office); email sales@nod32.com; website http://www.nod32.com/.

END NOTES & NEWS

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 3rd International Conference on Information Security takes place in Nicosia, Cyprus, 22-25 October 2003. Issues that will be addressed include: planning and implementing an IS strategy, considering the critical success factors, the business, legal and social issues; dealing with everyday intruders, keeping your network safe and investigating e-crimes. For details see http://www.cyprusinfosec.net/.

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 16–20 November 2003 in Las Vegas, USA. Educational programmes will take place 16–20 November, while the exhibition runs 17–20 November. See http://www.comdex.com/.

The Infosecurity.nl exhibition takes place 11–12 November 2003 at Jaarbeurs complex, Utrecht, Netherlands. For all the details, including information on how to participate, a list of exhibitors and floorplan, see http://www.infosecurity.nl/.

Infosecurity 2003 USA takes place 9–11 December 2003 at the Jacob K. Javits Convention Center New York, NY, USA. For information about the conference and exhibition, including online registration, see http://www.infosecurityevent.com/.

Kaspersky Anti-Virus for home users will be available in Germany from this month. *Kaspersky Labs* has reached an agreement with German software distributor *KOCH Media*, which establishes *KOCH Media* as the exclusive distributor of *Kaspersky Labs* boxed products in Germany. The first boxes of *Kaspersky Anti-Virus* for home users are due to appear on German shelves at the beginning of October 2003. See http://www.kaspersky.com/.

F-Secure Corporation* has released *F-Secure Anti-Virus Client Security, a centrally managed solution integrating anti-virus, distributed firewall, application control and intrusion prevention. For the details see http://www.f-secure.com/.

***Sybari Software* has added a sixth scanning engine to its *Antigen* product** for *Microsoft Exchange*, *Microsoft Sharepoint* and *Lotus Domino* environments. The *VirusBuster* engine becomes the sixth engine to be included in the *Antigen* product range, alongside *Norman Data Defense*, *Sophos*, *Kaspersky* and two *Computer Associates* engines from their *E-trust* suite. For more information see http://www.sybari.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