AUGUST 2004

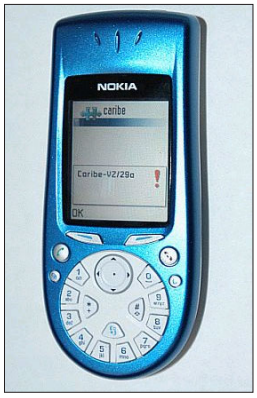
virus BULLETIN

The International Publication on Computer Virus Prevention, Recognition and Removal

CONTENTS

ISSN 0956-9979

IN THIS ISSUE



VIRUS CALLING!

It has been a long time coming, but in June 2004 the first worm arrived that spreads from mobile phone to mobile phone – bringing with it some new challenges for natural infection testing. Péter Ször and Peter Ferrie have the details of SymbOS/Cabir.

page 4

TURN IT OFF!

Virus notification emails, once useful, have become a bane, exacerbating the already overwhelming virus problem. David Ensign explains what went wrong and why he urges anyone with any type of automatic virus notification to turn it off.

page 12

COMPARATIVE REVIEW

Find out which of the 10 products for *NetWare* on test this month made the grade to gain a VB 100% award.

page 14

vbSpam supplement

This month: anti-spam news & events; automatic author identification for spam; ASRG summary.

'Is Mac OS ... magically protected from all past, present and future malware?'

David Harley
NHS Information Authority, UK

DIRTY MAC BRIGADE REVISITED

The Apple-using community is used to being stuck in something of a viral backwater, and has grown to think of viruses as something that happens to *Windows* users, who deserve everything they get. But is Mac OS, like *Linux*, magically protected from all past, present and future malware by its presumed superiority in interface and security model?

I like Macs, and continue to use them, sometimes. I appreciate the fact that Mac-specific malicious code rarely crosses my radar, and that I can, at need, navigate networks knowing that I'm temporarily immune to PC-specific unpleasantness. I hope that those who are fortunate enough to have unrestricted use of the computer they love most in the world continue to live their hobbit-like existence at a safe distance from the doings of dragons and orcs. However, wishful thinking and evangelism are not a sufficient defence against Nazgûl.

When things go wrong in the Shire – sorry, in the world of Macs – the implications may reach wider than the Mac community. In the mid-90s, Mac users who believed that the (now extinct) freeware package 'Disinfectant' would deliver them from all evil suddenly and dramatically became a major vector for macro virus dissemination. In 1998, AutoStart spread fast and far and caused serious damage to data: the SevenDust family spread less widely but also caused serious damage – however, these had little impact beyond the Mac community, though AutoStart could affect *NT* services under some circumstances. A minor change in the *Microsoft Office 2001* document format caused a number of virus-specific scanners to miss macro viruses in documents saved in that format – PC and Mac scanners were affected. Since then, malware issues have tended to be minor and localised – a proof-of-concept mass mailer (Mac.Simpson), an interesting Trojan experiment with a manipulated resource fork (MP3Concept), a trickle of destructive Trojans. Is there a pattern here? Not really, except that it does seem that every time the Mac community gets too complacent, *something happens*.

Are Macs intrinsically safer than PCs? A trawl through the Apple security site or third party security alerts services indicates a constant trickle of security vulnerabilities detected – often by third party researchers – and patched, but that's the world we live in. Some of these vulnerabilities could be exploited by virus writers and other black hats, but it doesn't seem to be happening much. Is that because the platform is bullet-proof, or because few care to shoot in that direction? Are Mac users, especially SOHO and home users, really better at keeping up with patches than their PC-using equivalents?

In principle, Macs are probably as vulnerable as they've ever been. OS X could be described as the traditional cuddly Mac interface to a flavour of Unix, and it's very well integrated. Many 'classic' Mac applications continue to work very well, and it would be rash to assume that old-style Mac viruses or contemporary analogues would not work equally well (though some anti-virus software seems to struggle a little in the OS X environment). But there is also a whole range of Unix-generic vulnerabilities and vectors that could be exploited. While the effects of such exploits should be mitigated by good administration (obvious stuff like patching, not logging in routinely as root, anti-virus maintenance, and so forth), I see no indication that most Mac (or *Linux*!) users are any better at such practices than most *Windows* users. Perhaps corporate PC users are better catered for in that there is a wider range of proven third-party protection and a solid shared knowledge-base of good practice. PC network administrators are generally aware of the fragility of their defences: they lock down desktops and move as much protection as possible to the corporate perimeter. Non-corporate computer users are a different can of worms. But *any* computer user fixated on the invulnerability of their OS of choice is prime social engineering fodder.

PROBATION FOR MAGOLD AUTHOR

Last month saw the sentencing of a Hungarian teenager to two years' probation for creating the Magold virus (see *VB*, August 2003, p.4).

Teenager 'László K.' was convicted of unauthorized use of computer systems. Although the teen was sentenced to one year in a juvenile prison, Veszprem City Court commuted the sentence to two years' probation and ordered him to pay 500,000 forints (approx. US$2,400) in court costs. According to a Hungarian newspaper, the teenager told the court that he created the virus to reassure himself that he had some skills after failing several subjects at his high school. However, blunders such as including his date of birth, most of his name and his postcode in the virus code (see *VB*, August 2004, p.4 for details) led to his arrest.

Also announced recently was the arrest, by the Finnish Central Criminal Police, of a man accused of creating and distributing the VBS/Lasku virus earlier this year. Unremarkable in most respects, VBS/Lasku's most unusual feature is that it spreads by sending email messages in Finnish – a fact which would, undoubtedly, have contributed to its lack of success on a global scale if it didn't already crash upon attempting to spread.

BUMPER QUARTER FOR SYMANTEC

Symantec has announced record profits for its fiscal first quarter of 2005. The AV company reported a profit of $131m on sales of $577m – profit having more than doubled since the same quarter last year. *Symantec* chiefs said some of the company's success could be attributed to the fact that the appearance of Sasser earlier this year sparked interest in its consumer product. CFO Greg Myers said: "The outbreak of Sasser had a powerful impact on our consumer channel, but enterprise [revenue also] grew by 29 per cent, which is better than our peer group."

Myers expected profits to be affected slightly this quarter by increased spending, which includes a recruitment drive across the company and another bout of acquisitions. Adding to its reputation as something of a 'shopaholic' among AV vendors, *Symantec* purchased anti-spam startup *TurnTide* last month. The AV vendor is reported to have paid $28 million cash for the acquisition of the small firm, which was spun off from privacy protection consultancy *ePrivacy Group* just six months ago, and sells a router-based email filtering technology. The purchase came just three weeks after *Symantec* completed its acquisition of anti-spam firm and email filtering company *Brightmail*. According to *Symantec*, *TurnTide*'s technology is intended to become part of a multi-tier anti-spam line-up that will include *Brightmail* and other *Symantec* products.

Prevalence Table – June 2004

Virus	Type	Incidents	Reports
Win32/Netsky	File	215,228	82.41%
Win32/Bagle	File	29,557	11.32%
Win32/Zafi	File	6,508	2.49%
Win32/Sober	File	4,953	1.90%
Win32/Dumaru	File	1521	0.58%
Win32/Klez	File	491	0.19%
Win32/Lovgate	File	317	0.12%
Win32/Bugbear	File	282	0.11%
Win32/Mydoom	File	266	0.10%
Win32/MyWife	File	206	0.08%
Win32/Swen	File	190	0.07%
Redlof	Script	183	0.07%
Win32/Funlove	File	181	0.07%
Win32/Mimail	File	161	0.06%
Win32/Valla	File	113	0.04%
Win32/Fizzer	File	112	0.04%
Psyme	Script	82	0.03%
Win32/Hybris	File	72	0.03%
Win95/Spaces	File	72	0.03%
Win32/Parite	File	71	0.03%
Win32/Yaha	File	55	0.02%
Win32/Magistr	File	54	0.02%
Win32/Sobig	File	48	0.02%
Win32/Sasser	File	46	0.02%
Win32/Nachi	File	43	0.02%
WYX	Boot	42	0.02%
Win32/Gibe	File	33	0.01%
Win32/Lovsan	File	33	0.01%
Laroux	Macro	29	0.01%
Win32/BadTrans	File	28	0.01%
Win32/Korgo	File	26	0.01%
Fortnight	Script	24	0.01%
Others[1]		151	0.06%
Total		261,178	100%

[1]The Prevalence Table includes a total of 151 reports across 37 further viruses. Readers are reminded that a complete listing is posted at http://www.virusbtn.com/Prevalence/.

VIRUS ANALYSIS 1

CABIRN FEVER

Peter Ferrie and Péter Ször
Symantec Security Response, USA

It has been a long time coming, but in June 2004 the first worm arrived that spreads from mobile phone to mobile phone: SymbOS/Cabir. Fortunately, due to the fact that the worm uses a specific user-interface component, it is restricted to *Series 60*-based mobile phones.

BLUETOOTH

Cabir spreads using the Bluetooth wireless networking technology. The name of the technology comes from a translation of the name of the tenth century Danish king Harald Blåtand, or Bluetooth. Some say that it was Blåtand who united Denmark and Norway, and Bluetooth which unites the rest of the world. That would be for large values of 'unite' or small values of 'world', though, given the very limited range of Bluetooth.

ALL SIS-TEMS GO!

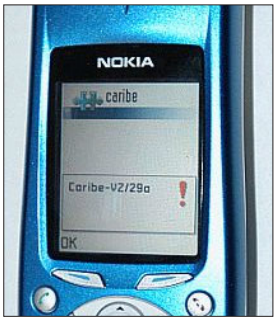
Cabir arrives as a .SIS file. A .SIS file is an installation package that contains files and/or scripts, and is processed by the Installation Manager that is part of the *Symbian* operating system.

When the Cabir .SIS file is executed, the Installation Manager will extract and place the worm files (CARIBE.APP, FLO.MDL and CARIBE.RSC) into the '\SYSTEM\APPS\CARIBE' directory.

The Installation Manager also creates a file named '\SYSTEM\INSTALL\CARIBE.SIS', which contains only information about how to remove the installed application. The .SIS file is configured so that the Installation Manager will then run the extracted 'CARIBE.APP' file. This application runs on the ARM series of processors.

HELLO WORLD

When the 'CARIBE.APP' file is executed, it displays a message announcing its presence. The message is 'Caribe-VZ/29a' in the .A variant, and 'Caribe' in the .B variant. Once the user clicks 'OK', the worm waits 10 seconds before proceeding.



Cabir begins by checking the filename of the currently running file. If the filename is not 'CARIBE.APP', running from the directory C:\SYSTEM\SYMBIANSECUREDATA\CARIBESECURITYMANAGER' (note the hard-coded 'C:', which is the default drive but is not always used), the worm will create that directory, and copy itself there as a file named 'CARIBE.APP'.

The worm also copies the .rsc file to the same directory, as a file named 'CARIBE.RSC'. If there is a failure during the copying of the .rsc file, the worm deletes the 'CARIBE.APP' file that has just been created. Such a failure will prevent the worm from running by default, if the device is restarted.

Finally, the worm creates a directory named 'C:\SYSTEM\RECOGS', and copies to this directory the file named 'FLO.MDL'.

MIMETIC BEHAVIOUR

The 'FLO.MDL' file is a MIME recogniser. MIME recognisers that are present in the 'RECOGS' directory are called whenever applications are launched, and are used to prepare the environment for an application to run.

The 'FLO.MDL' file simply runs the 'CARIBE.APP' file from the 'CARIBESECURITYMANAGER' directory, rather than from the 'APPS' directory. This means that even if a user uninstalls the CARIBE application, the worm will continue to run. Additionally, files under the 'SYMBIANSECUREDATA' directory are not visible by default to users unless File Manager is installed, which is able to show these files.

The more recent models of *Series 60* devices do not allow MIME recognisers to run files in directories other than the 'APPS' directory.

After copying the necessary files, the worm creates a new .SIS file, using the .SIS file header that it carries, and the files that it has copied. This step is necessary because a .SIS file is usually deleted by the Installation Manager after an installation has completed.

FIRST CONTACT

Cabir attempts to spread using a three-stage approach. The first stage searches for Bluetooth-enabled devices, and attempts to connect to the first one that is found, regardless of the type of device (i.e. even a printer or a mouse will be attacked if they are in range).

The second stage sends the 'CARIBE.SIS' file, and the third stage disconnects from the target device. Immediately after disconnecting from the target device, the worm runs the first stage again. The worm will connect to the same device

again and again, for as long as it is in range. The cycle continues for as long as the worm application is allowed to run.

The first-stage search for Bluetooth-enabled devices causes a significant drain on the battery of the mobile phone – however, Bluetooth support can be switched off by the user, since the worm does not switch it on (though, by default, Bluetooth support is not enabled).

CLICK ON EVERYTHING

Cabir requires several interactive steps on the part of the recipient in order to execute. The first step is to accept the incoming connection request from another user (this would soon become a great many requests while the two devices remain in range of each other).

Having accepted the request, the recipient is presented with a warning that the supplier cannot be verified. However, this warning message is displayed by any application that originates from anywhere other than *Symbian*, even if that application is signed.

If the recipient elects to continue anyway, a final prompt is displayed that asks whether the recipient wants to install the application. Only if this prompt is accepted will the worm be installed and executed.

FLYING CIRCUS

Cabir introduces new problems to natural infection testing. Normally, it is sufficient to walk into a secure zone and work on an isolated network. In an attempt to test Cabir, one analyst tried to find the closest emergency evacuation bunker; another suggested running out into the middle of a deserted park with no one nearby. Although sending files through well built walls does not seem to work reliably, the wireless security of the virus labs needs to be prepared for wireless devices that use stronger signals.

What will be next? A mass mailer using MMS? A downloader using SMS? Ring, ring your virus is calling!

SymbOS/Cabir	
Size:	11,944 bytes (.A), 11,932 bytes (.B).
Type:	Mobile phone worm.
Payload:	Phone battery drained by search method.
Removal:	Delete the referenced files.

VIRUS ANALYSIS 2

MOSTLY HARMLESS

Peter Ferrie and Frédéric Perriot
Symantec Security Response, USA

The LSASS vulnerability of *Microsoft* security bulletin MS04-011 affects *Windows 2000* and *XP*, the two most widespread *Microsoft* operating systems today. It is a stack overflow, hence easily and reliably exploitable – and *eEye* was kind enough to provide the world with thorough documentation of the possible exploitation vectors.

Following in the path of previous high-profile vulnerabilities, the LSASS bug was quickly targeted by proof-of-concept exploits, themselves reused in worms including W32/Sasser.A. Despite the publicity that surrounded Sasser due to its immediate success following its appearance (30 April 2004), this was not the first worm to make use of the vulnerability: some LSASS-exploiting Gaobot variants had surfaced about a week earlier. However, it was the automated infection of new systems that was the decisive factor in making Sasser more widespread.

BISTROMATHICS

Sasser infects new systems by exploiting one of the many vulnerabilities announced in the MS04-011 bulletin. The vulnerability is related to a stack buffer overflow in the file lsasrv.dll, which is normally loaded as part of the lsass.exe process (Local Security Authority Subsystem Service).

In order to find new victims, Sasser scans random IP addresses for vulnerable machines listening on port 445/tcp. Once such a machine is found, it attempts to exploit the LSASS vulnerability by sending a specially crafted RPC request to the LSASS named pipe on the machine. Upon successful exploitation, shell code is injected into the lsass.exe process, which executes a shell (cmd.exe) and binds it to a TCP port. The attacking instance of the worm then connects to this port and sends commands to the shell. These commands download and run the main worm executable on the newly infected system.

The worm download is carried out through FTP, using the default *Windows* ftp.exe program on the client side (victim). On the server side (attacker), Sasser implements its own crude FTP server, which listens on a non-standard TCP port. The infection scheme of Sasser is very similar to that of W32/Blaster (see *VB* September 2003, p.10), with the exception of using FTP instead of TFTP as the main transmission protocol. Worms get more reliable!

Once it is running on a new machine, Sasser installs itself in the Windows directory under the name ‘avserve.exe’ and registers itself in ‘HKLM\...\Run’ as the value ‘avserve.exe’,

in order to run on *Windows* startup. Then Sasser simply tries to infect new systems. There is no intended payload, time-triggered routine, or anything other than replication code in this worm.

INFINITE IMPROBABILITY DRIVE

Sasser generates target IP addresses using three different methods: completely random IPs are used 52 per cent of the time; random IPs located in the same /16 network as the host are used 27 per cent of the time; and random IPs located in the same /8 network as the host are used 21 per cent of the time. This method of skewing probabilities towards nearby hosts was used in W32/Welchia (see *VB* October 2003, p.10). The aim is to increase the probability of hitting vulnerable hosts, on the assumption that nearby machines suffer from the same misconfiguration problems.

The network scanning speed per attack thread is a maximum of four attacks per second, and Sasser spawns 128 attack threads running in parallel.

VOGON POETRY

In addition to the scanning threads, Sasser creates an extra thread devoted to listening for incoming connections on its FTP server port, 5554/tcp. Each incoming FTP connection is then serviced by a newly spawned thread.

Despite its extreme simplicity, the FTP server code in Sasser is buggy. All the code does is reply to five basic FTP commands – 'USER', 'PASS', 'PORT', 'RETR' and 'QUIT' – with some canonical answers, to please the ftp client of the other side of the connection, and serve the worm executable over an FTP data channel.

In the one command requiring a minimal amount of parsing, 'PORT', there lurks a buffer overflow due to the use of a strcpy() call (more on this later).

Sasser creates an embryonic log file in 'c:\win.log'. This may, originally, have been intended as a list of compromised systems, but due to what appears to be a bug, it remains always one line long. The file contains the IP address of the last system successfully infected, and a counter indicating how many systems have been infected from the local machine, in total, since the worm started running.

BABEL FISH

Sasser exploits one vulnerability, but it really makes use of two different exploits, depending on the platform that it is attacking. We shall refer to them as the 'short-form' and 'long-form' exploits. Moreover the 'long-form' exploit has two variants, using two different trampoline addresses.

In order to determine which *Windows* platform it is attacking, the worm fingerprints the remote target system by establishing a NULL session with it, and checking the Native OS field of a session setup response packet. Based on the contents of the Native OS field, '5.1', '5.0', or neither of these two strings, Sasser picks the short-form, long-form (first variant) or long-form (second variant) exploit, targeted respectively at *Windows XP*, *Windows 2000* and unidentified systems. The NULL session packets produced by Sasser seem to originate from a regular *Windows 2000* system, because the author of the exploit captured sample traffic and simply replayed it in the exploit code.

The mode of operation of the short-form exploit, used against *Windows XP*, is to hijack a return address. The trampoline address used in this case points to a 'call esp' instruction located in the address space of the lsass.exe module itself. (This is contrary to the comment in the publicly available source code of the exploit, which mentions it as a 'jmp esp'.)

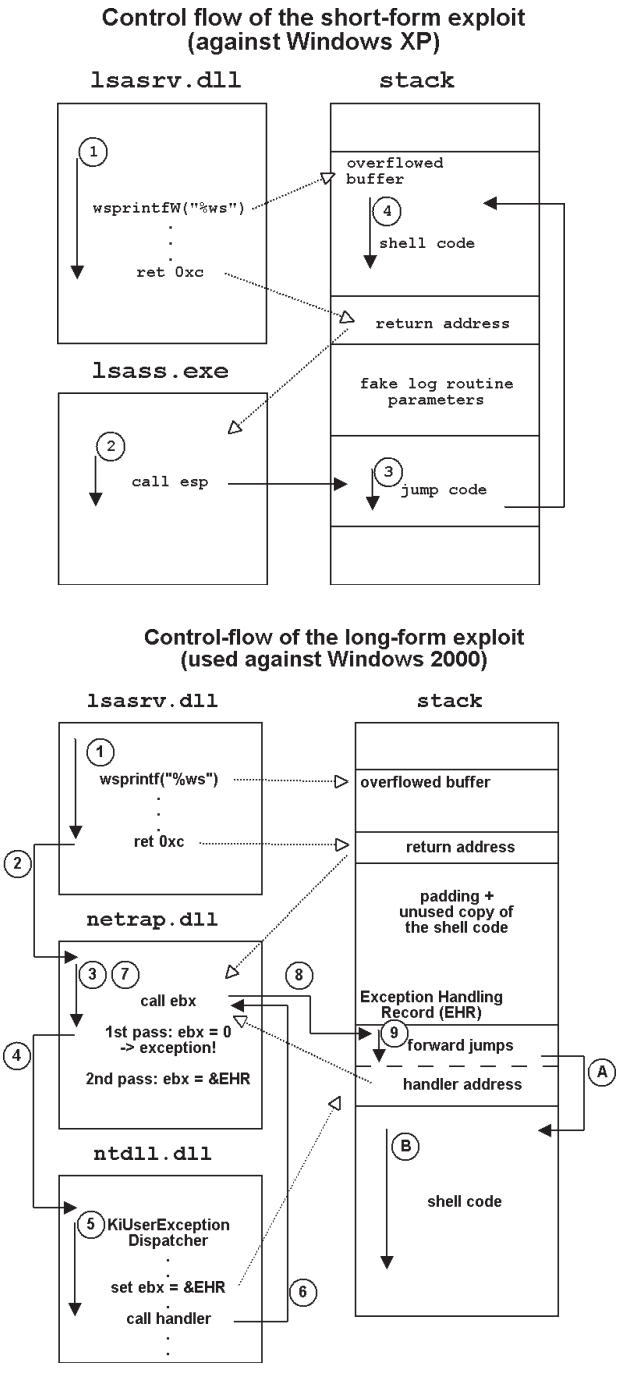
The long-form exploit is more complicated: it attempts to hijack both a return address and an exception handler on the stack. The trampoline address used against unidentified systems points to a 'call ebx' instruction in netrap.dll. The one used against *Windows 2000* systems points to a 'jmp ebx', according to the author of the exploit – but we have not been able to verify this information on our test platforms. The combined hijacking of a return address and an exception handler is probably designed to improve the reliability of the exploitation. In our tests, however, only the exception handler part was needed, and the sole purpose of hijacking the return address was to trigger an exception. (For more detail on the control flow of the exception handler hijacking trick, see *VB*, September 2001, p.4).

The layout of the attack buffers and the control flow of the exploits are depicted opposite.

HEART OF GOLD

Regardless of the exploit flavor, the attack starts by opening a connection to port 445/tcp of the target system, then going through the same protocol negotiation and session setup as in the fingerprinting. Next, the '\lsarpc' named pipe is opened on the remote system, and an RPC request is made against the LSA_DS (Directory Services) interface.

In the short-form exploit, Sasser sends an approximately 2kb attack buffer to the LSA_DS interface. The RPC request is small enough to fit in one RPC fragment, and the entire request is carried out with a single named pipe transaction. In the long-form exploit, the attack buffer is about 7kb long and the RPC request is split into two fragments.



As for the fingerprinting, the network traffic generated by the Sasser exploits is more akin to a replay than truly synthesized. The author of the HouseOfDabus exploit, which is reused in Sasser, is likely to have obtained it by sniffing traffic from a first-generation exploit that relied on a modified version of netapi32.dll, which was tampered with to allow an extra parameter in the signature of the DsRolerUpgradeDownlevelServer() function. This function employs the LSA_DS RPC interface for legitimate purposes, but normally operates against the local system only. The extra parameter in the version that has been tampered with identifies a machine, thus allowing remote exploitation.

The effect of the malformed request to the LSA_DS interface is a stack buffer overflow in the DsRolepDebugDumpRoutine() logging function of lsasrv.dll. The vulnerable function is normally used to write information to a file called 'DCPROMO.LOG', located in the '%windows%\debug' directory. It employs a 2kb stack buffer to hold log file lines, and it exercises no bounds-checking prior to using a sprintf('%ws') function to fill the buffer. By providing an over-long value corresponding to the '%ws' parameter, Sasser controls the return address and the rest of the stack after the buffer, which allows the execution of arbitrary code. Sasser relies on this to execute its shell code.

The exact sprintf() function in the buggy log routine varies among operating systems and service packs. For *Windows 2000*, the function is imported from msvcrt.dll, and is sprintf() in SP0, and vsprintf() in SP4. For *Windows XP*, the function is imported from user32.dll, and is wsprintfW() in SP0 and wvsprintfW() in SP1a.

The use of *Unicode* and ASCII functions (with or without the leading 'w' in the function name) explains the need for a long-form exploit and a short-form exploit providing more or fewer bytes in the '%ws' parameter for different platforms.

ZAPHOD WAS HERE

Surprisingly, Sasser performs the attack twice for each target machine. The likely explanation for this behaviour is also related to the kind of sprintf() function used in DsRolepDebugDumpRoutine(), more specifically to the use of the user32.dll implementation of sprintf() on *Windows XP* platforms.

The user32.dll implementation of the sprintf() functions has a limit of 1024 characters (not bytes – *Microsoft*'s documentation is not quite correct) that it will place in the destination buffer. Once the limit is reached, no further data are placed in the buffer. On the first exploitation attempt of a *Windows XP* system, when the logging routine is called from DsRolerUpgradeDownlevelServer(), this limitation is encountered because the size of the '%ws' parameter combined with the log line header exceeds 1024 characters.

As a result, the stack buffer is missing a final carriage return and a flag is set, indicating that the next bit of log information should simply be appended to the current line. On the next exploitation attempt, the flag is checked and the long '%ws' parameter is copied to the stack buffer without a

header. The '%ws' parameter alone fits in the buffer, no stack overflow occurs, and 'eventually' (readers are welcome to contact us if they have questions about this) the aforementioned flag is reset, allowing exploitation again.

The net effect is that exploitation of *Windows XP* systems works only every other time – at least under normal conditions. This explains the double attempt at exploitation by Sasser. The author might have assumed that the same condition could occur while attacking *Windows 2000* machines, but we have not observed this in our tests.

Once the lsass.exe process is coerced into running the shell code injected by the worm, the code binds to port 9996/tcp, accepts a connection from the attacker and runs a 'cmd.exe' shell as the SYSTEM user. The commands sent to the shell cause the worm executable to be FTP'd to a file whose name starts with four to five random digits followed by '_up.exe'.

DON'T PANIC!

Despite the obvious success of its spreading mechanism, Sasser suffers from a major limitation: it uses the wrong exploit parameter when it attempts to infect English versions of *Windows 2000* systems. Tests in the lab show that the trampoline address used by Sasser against English versions of *Windows 2000 Workstation*, *Server* and *Advanced Server*, *SP0* and *SP4*, does not correspond to a branch, but to another instruction (a locked 'mov') that causes an exception when reached (including from the exception record hijacked by Sasser, which leads to a crash).

We believe this behaviour is related to the single-byte vs. double-byte character platform issue, and that the Sasser exploit targeted at *Windows 2000* systems works only against double-byte character platforms. This is corroborated by reports from the field: of all Sasser submissions received by *Symantec* from the 'C:\WINNT' directory (the default installation directory for *Windows 2000*, whereas *Windows XP* uses 'C:\WINDOWS' by default), the vast majority originated from machines located in Taiwan and China. The source of the other submissions was unclear, but the names of the users suggested they may have been double-byte character platforms as well.

Thus, Sasser's exploit is effective only against *Windows XP* and some versions of *Windows 2000*. It fails against *Windows 2003 Server* (in fact, the buggy function is not even called).

DISASTER AREA

As in the case of W32/Blaster and other exploit-based worms, the end result of missed exploitation attempts against vulnerable systems is often a crash of the attacked process. The crash – in lsass.exe in the case of Sasser – manifests itself as an error message box warning the user that the system will be shut down. The author of the worm tried to call AbortSystemShutdown() from the main worm executable to prevent this suspicious behaviour, apparently overlooking the fact that the main worm code does not run at all if the exploit fails! On the other hand, if the worm is running successfully on a system that is then incorrectly compromised, the error message box might not appear.

If the exploit succeeds, the shell code terminates with an ExitThread() call after spawning the shell. This is clean enough by itself to ensure that the lsass.exe process keeps running, and makes the use of AbortSystemShutdown() unnecessary.

RESISTANCE IS FUTILE

From the point of view of Network Intrusion Detection, Sasser has one interesting feature: it sends the FTP data and the shell commands byte-by-byte, which results in the network traffic it sends being split into TCP segments starting at unpredictable boundaries. This may have been designed as a way to evade IDS products which do not have the capability to reassemble TCP streams. It may also have been accidental, since no such care is taken when the RPC attack buffer is sent. Nevertheless, the potential exists to mislead some IDS systems.

It was not long before a jealous contender attempted to take advantage of the vulnerability in Sasser's FTP server code. W32/Dabber, which appeared on 14 May 2004, does just this.

Soon a whole new branch of the security industry will appear, specializing in detecting exploitation of worm vulnerabilities: "*Protect your Sasser-infected machines with JamScan. JamScan not only stops Dabber, it also protects you against future worms exploiting the same flaw!*"

W32/Sasser.A

Aliases:	W32.Sasser.Worm, WORM_SASSER, Worm.Win32.Sasser.
Size:	15,872 bytes.
Type:	Internet worm.
Exploits:	LSASS vulnerability, MS04-011, CAN-2003-0533.

VIRUS ANALYSIS 3

MORE PATRIOT GAMES

Gabor Szappanos
VirusBuster, Hungary

In the concluding remarks of last month's Zafi.A analysis (see *VB*, July 2004, p.6), Tibor Marticsek and I predicted that we would be likely to see further variants of this virus in the future. That we knew for sure. What we didn't know was that it would happen so soon.

The timing of this virus cannot be ignored. The first variant appeared in the second half of April 2004, immediately before the date when Hungary became a member of the European Union. The second variant appeared in June 2004, immediately before the European Parliament elections – which suggests at least partial political motivation. This is supported by the fact that the second variant seems to be unfinished. The tampering check which was present in variant A is still present in variant B, but it is not executed. Also, Zafi.B has a hidden political message which is never displayed – it is as if the worm had to be finished in a hurry to be ready before a deadline.

The major improvement in the worm is that it is capable of spreading in different languages, and is not limited to Hungarian addresses only. The result is that, while Zafi.A topped only the Hungarian virus prevalence charts, Zafi.B was the most prevalent virus according to the statistics of most of the global anti-virus companies in June 2004.

ZAFI.B

The first sample of Zafi.B was received on 10 June 2004. However, our web server logs indicate that the first samples may have been activated the day before (the worm performs a DDoS attack against our website, www.virusbuster.hu). Like the previous variant, Zafi.B was written in assembly language, but it was packed with FSG as opposed to UPX which was used by variant A.

When it runs, Zafi.B copies itself to the %System% folder as .exe and .dll with random names. It creates several .dll files with random names where it stores the email addresses. In the registry key 'HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run' it creates the entry:

```
_Hazafibb-=%System%\-created .exe file name
```

The virus data is stored under the registry key 'HKEY_LOCAL_MACHINE\Software\Microsoft_Hazafibb'. Variant A used the variable 'Hazafi', this one uses 'Hazafibb', which is the comparative form of 'Hazafi'. I have the feeling that we will soon see the superlative form (which should be 'Leghazafibb').

Zafi.B scans folders, and copies itself to folders whose names contain the 'share' or 'upload' string, using one of the following names:

```
winamp 7.0 full_install.exe
Total Commander 7.0 full_install.exe
```

It disables the execution of several anti-virus and firewall programs. However, this is not done in the usual way, simply by stopping the processes. Instead, it searches for programs that contain the strings 'firewall' or 'virus' and overwrites the program files with itself after stopping the process associated with the program.

Zafi.B also disables the use of the critical system utilities with names matching regedit*, msconfig* and task*. This is also achieved in a somewhat unusual manner: the worm opens these programs with the CreateFile API, thus the execution attempt will fail.

The worm scans for email addresses in .htm, .wab, .txt, .dbx, .tbb, .asp, .php, .sht, .adb, .mbx, .eml and .pmr files, and sends infected emails to those addresses. The worm uses different messages depending on the domain extension of the email addresses. The recognized domain extensions are: .hu, .sp, .ru, .dk, .ro, .se, .no, .fi, .lt , .pl, .pt, .de, .nl .cz, .fr, .it, .mx and.at. It also uses a different set of mail server prefixes for each recognized country.

At the end of execution the virus deletes the following files (if they are running, it terminates the appropriate processes first): fvprotect.exe, winlogon.exe, jammer2nd.exe, services.exe. These filenames are used by Netsky variants.

The worm performs a denial of service attack against the following websites: www.parlament.hu, www.virusbuster.hu, www.virushirado.hu and www.2f.hu.

During the attack it sends empty GET requests to these websites in an endless loop. There were some slowdowns in accessing these websites during the highest infection time, however none of them were brought down.

This attack gave us a chance to estimate the total number of computers infected with Zafi. On Monday 13 June 2004 between 06.00h and 15.46h (CEST) we counted 105,926 different machines (IPs) attacking our web server. These made up to 100,000 attempts per minute. By the end of that week, the number of different IPs decreased to about 30,000 during the same time frame.

CONCLUSION

Will we see more variants of Zafi? Surely. Zafi.B seems to be an unfinished project, which indicates that the development is ongoing. If not earlier, then we may see more of Zafi in 2006, to coincide with the next elections in Hungary. Real patriots cannot keep quiet in testing times.

OPINION

THE END OF CYBERCRIME?

Eddy Willems
NOXS (formerly Data Alert) and EICAR, Belgium

I'm getting old. The idea of my 12-year-old son and his friends playing the latest version of 'Counterstrike' at a LAN party gives me an uneasy feeling. It is a very bizarre sight to see young people staring silently at TFT screens, as if they are in some sort of trance. After visiting a number of local organised LAN parties I discovered that it is not only gaming and the copying of games that interest our children. Now, it seems, their interests extend to the hacking and cracking of PCs, websites and so on.

MEGA LESSONS

My wife is a police officer, and a so-called 'MEGA' officer. MEGA is the European version of the US-based DARE (Drugs Abuse Resistance Education) project. MEGA officers visit local schools giving MEGA lessons to 12-year-old school children. The project has been created to raise children's awareness of the problems associated with drugs, alcohol, violence, and so on. One of the additional aims of the Belgian project is to make children aware of the dangers that reside on the Internet.

WHAT DO CHILDREN KNOW?

I decided to do some research to try to find out what really goes on in the minds of young people. I asked a number of MEGA officers and school teachers about the levels of computer security awareness they find in the children they meet. I also posed a set of basic questions about computer security to a small group of children. A selection of their responses is given beneath each question here:

1. *Do you know what a computer virus is?*
   "No"
   "It eats your emails"
   "It wipes out everything"
   "The computer is sick"
2. *Have you ever been infected by a computer virus?*
   20% said "Yes'"
   10% said "No"
   70% said "I don't know"
3. *What are the effects of a virus?*
   See answer 1. However, it became clear that none of the children really seemed to know what viruses do – none of them mentioned self-replication.
4. *Do you click on every link in an email and open every attachment you receive?*
   99% said "Yes"
5. *Do you use an 'easy' password, such as your first name, birthdate, etc.?*
   80% use a *blank* password if possible
   16% use an easy password
   4% use a difficult password
6. *How can you surf on the Internet safely?*
   "I surf to kids' sites" [laughing]
   "Isn't the Internet safe?!"
   "I use Google"
7. *What do you think of virus-writing or hacking?*
   "Cool"
   "Dangerous for your health"
   "Oh my father does it all the time…"

Of course this is not a scientific poll, and it reflects the ideas of only a small selection of children in one region of the world. However, together with other feedback and after discussions with MEGA officers, I arrived at several conclusions and opinions.

CONCLUSIONS AND OPINIONS

Children do not seem to know what computer security is. Some of them even find the idea of becoming a hacker or a virus writer 'cool'. Although some families use parental control mechanisms to secure their home computer networks, many children know how to bypass these mechanisms.

Generally, it seems that our children's knowledge of ethical computer behaviour and good 'netiquette' are a long way off target.

A suggestion as to how we may begin to influence students and young people is by using societal control. An example of how this has worked in the past is with the issue of drink-driving.

At one time, drinking and driving was a personal choice, but as society witnessed some of the consequences of the combination of the two activities, we began to pass laws which restricted such behaviour. Initially there was some resistance to these laws – people saw them as an infringement on their rights. However, as the laws became more widely accepted, people began to refuse to drink and drive on the principle that it is 'wrong' to do so.

Policy makers and law makers are very aware of this form of societal control. However, they are less aware of the societal structure of 'cyberspace', and for this reason there is the danger that the laws they make will not create the desired ethical model, and conversely will create a backlash or revolutionary movement. By taking time to develop realistic policies and effective laws, it is possible we can avoid such a reaction.

The speed with which global electronic communication is developing has brought with it an enormous benefit to all those fortunate enough to be able to exploit it. However, it has also brought opportunities to those who are willing to abuse it.

The way in which it has introduced relative and absolute anonymity for its users may encourage acts which would otherwise have appeared to be too risky to the perpetrator. Its very nature may encourage various kinds of anti-social activities, ranging from innocent pranks through serious malicious damage to data and individuals, and downright criminal fraud.

As a result of the fact that many of its principle users are relatively young, or people who may be impressionable or unprincipled, an ethos has developed in the Internet community, in which it is 'cool' to be an outlaw. Moreover, the inherent power embodied in being able to control the 'system' is potentially irresistible.

Resources that would enable us to emphasize and integrate ethical computing behaviour may provide a stabilizing influence. Our computing environments are very vulnerable regarding distribution of information – after all, it is what they were designed to do.

If we want to change people's behaviour and reduce the attractiveness of becoming a virus writer or hacker, we must start ethical computer education at a much earlier age. I think the way forward is to recognize the different factors introduced by computer technology – factors we have long ignored. If we don't, the technology may ultimately be self-destructive.

So why not incorporate this information into the MEGA lessons or the DARE project and start educating our children at a much earlier age?

This research project is definitely not finished and I would like to put more psychological elements into the project with the help of teachers, sociologists and psychologists. The research will be re-evaluated next year. In the meantime, I remain open to suggestions and comments from anyone who has experience in this field.

[*Readers interested in contributing to Eddy's research project can get in touch with him via the Editor of VB – email editor@virusbtn.com.*]

FEATURE

HELP! TURN OFF THOSE NOTIFICATIONS!

David Ensign
SAIC, USA

We have all seen it. User John Doe is working productively when he receives an email message from an outside agency:

```
The WebDevice SMTP Scanner at site.com detected
the virus W32/Netsky.p@MM in attachment file.doc.pif
from <john.doe@business.com>.

Contact your system administrator immediately.
```

Not unexpectedly, his reaction is to stop working and contact his support staff. He informs them he has a virus alert on his system.

By now, most support staff know that, because of spoofed addresses, these email messages are false reports, but maybe John Doe doesn't clarify that it was an email message and not a pop-up alert. By the time everyone sorts it out, the user has lost an hour of work, and support resources have spent an hour working with him. In any sizable organization, this happens dozens of times a day.

These notifications, once useful, have become a bane, exacerbating the already overwhelming virus problem. It started out as a noble, well-intentioned, and (initially) effective idea. With the advent of mass-mailing viruses (starting with the Melissa macro virus in March 1999, and soon followed by numerous variants, most notably the LoveLetter VBS worm in May 2000, Naked in March 2001, and BadTrans in July 2001), thousands of systems worldwide had become infected, each generating hundreds or thousands of infected messages.

In most cases, users were oblivious to the virus activity within their systems, and the lack of widespread virus protection had finally come home to roost. The cybersecurity community, however, saw an opportunity. With perimeter SMTP gateway or post office protections gaining popularity, many sites were intercepting the infected messages at the point of entry, preventing incursions. This also provided the potential to reply to the senders with a notification that their computers were infected. Unprotected users could be made aware of the compromise through these 'bounce-back' messages and hopefully would take steps to rectify the problem, eliminating sources.

Those early mass-mailers all had one thing in common: they utilized *Outlook* as their email engine. Because of the ubiquity of *Microsoft* applications, especially in businesses, this was very effective, but *Outlook* had some constraints, the main one being that it required that any generated email could only originate from the user name of the specific user. In other words, the name in the 'From' address represented the actual user whose system initiated the message. As a result, any bounce-back was assured to return to the originating user who surely was infected. Essentially, the efficacy rate was 100 per cent.

It is likely that this mechanism was the primary reason that most early mass-mailing viruses flared briefly and then faded away as the population of users with infected systems were alerted to their situation and implemented protections. In fact, if you follow the course of prevalent viruses from 1999 through to May 2003 (the last time a non-spoofing virus – Magistr – was listed in the top 10 of *VB*'s prevalence table), these early mass-mailers, although superseding their non-mailing brethren, rarely exceeded their peak prevalence.

As Figure 1 shows, there were only two months (March and December 2001) when the number of encounters from *Outlook* worms exceeded the high-water mark for basic viruses (reached in March 1999), and in each instance there were steep declines in the succeeding month. No *Outlook*-based worm held the top position on the prevalence list for more than one month. A reasonable conclusion is that the ability to identify the infected party allowed for subsequent notification, leading to remediation and the elimination of a system as a source. In a short time, each virus died out.

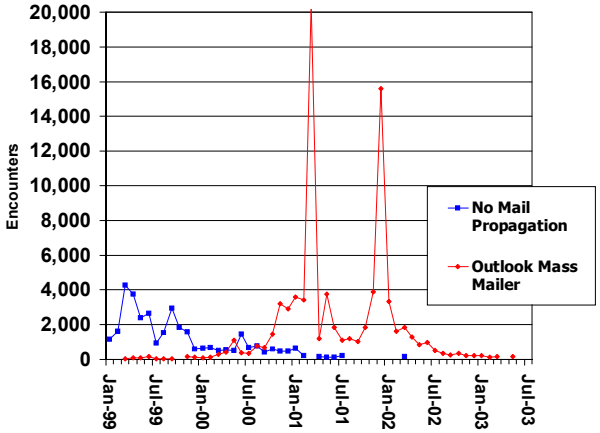


Figure 1: Comparison of Outlook-based worms versus viruses with no email propagation component (using VB statistics on encounters).

This changed at the end of 2000. In October 2000 MTX, while not a spoofer, did embed its own SMTP engine. The consequence of this was that the worms were no longer constrained by the rules of *Outlook*. In particular, they could use any sender address they wanted, and so spoofing took the first steps to dominance, soon to be adopted not only by virus writers but by spammers as well. Hybris was the first notable strain to include spoofing, albeit rudimentary (all messages were from 'hahaha'). It wasn't until SirCam in July 2001, that an effective spoofer arose. Today's worms harvest addresses from a wide variety of sources on an

infected system, arbitrarily picking some to be the senders. They also generate more mail than ever. The end result is that any user's address is likely to be used at some point.

The impact of this evolution was dramatic, essentially relegating bounce-back notifications – which had grown to be a primary countermeasure – to obsolescence. Any attempt to send a message back to the 'sender' was short-circuited, because the message rarely, if ever, found its way back to the actual infected originator. The immediate result was that the cybersecurity community lost the ability to notify infected users of their condition easily. Now infected systems could continue their actions unimpeded, their users once again oblivious.

With no mechanism to identify and alert these users, no rectification would occur, and virus activity could essentially continue *ad infinitum*. Viruses became harder to eradicate. Unlike the short-lived *Outlook* worms, SirCam was the most prevalent virus for seven out of nine months through March 2002; Klez for seven out of eight months in 2002; Sobig for three out of four in 2003; and Netsky for three straight months (and continuing) in 2004. These viruses are hard to find, and you can't kill what you can't find. Perhaps the only reason they stop is due to saturation.

These infected, spoofed emails produce several types of automatic notifications. One occurs when the harvested address turns out to be invalid. Ultimately, this will generate an undeliverable message from the receiving mail server sent back to the 'sender'. Another message may be generated because of policy blocks at SMTP gateways or mail servers, perhaps prohibiting the immigration of executable attachments. Because these notifications provide important service outside of the virus issue, eliminating them as a policy is not practical. Fortunately, most harvested addresses are legitimate and most gateways check for viruses before policies, so these invalid responses represent a small percentage of the bounce-back traffic.

The third message is the virus detection notification, which is far more common. These usually are generated by gateway or infrastructure virus scanners, and each one represents a successful interdiction of a virus incursion. In most cases, each product or site has created a personalized, tailored message, which is a key point, as it prevents the implementation of viable content policies to block these bounce-backs from reaching users.

But spoofing mass-mailers have been around for years now. Why, then, are bounce-backs an issue today? The answer is scale. Today's viruses gain worldwide saturation in days, infecting many systems quickly, and then generating massive amounts of contagions. As Figure 2 shows, these SMTP mass-mailers produce significantly more encounters as a class than any previous type.

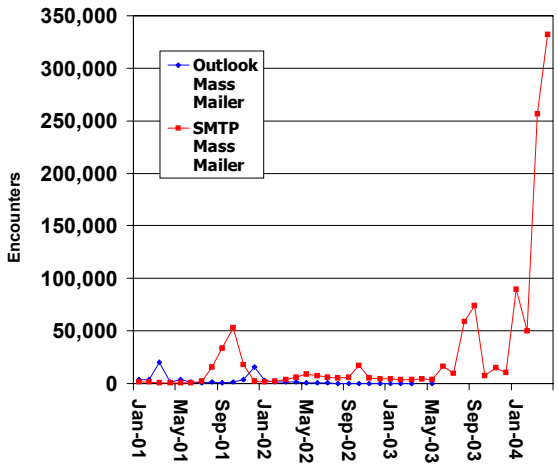


Figure 2: SMTP-based worms versus Outlook-based worms (using VB prevalence data).

The biggest *Outlook*-based worm (Naked) produced no more than 20,000 encounters in any month. The first major SMTP worm, SirCam, produced over 50,000. Outbreaks of over 50,000 monthly encounters now are common, and Netsky, with over 250,000 encounters in March and 300,000 in April 2004, represents a quantum leap. We now see more encounters in a month than we saw in the whole of 2003.

This means that many more messages are being intercepted and there is a greater possibility that a user's address is being used as a sender. The *MessageLabs* monthly report for May 2004 (see http://www.messagelabs.com/) showed that virus messages represented about 9 per cent of email traffic. With widespread activation of notifications, we are exacerbating the problem by generating nearly as much bounce-back email traffic, with no useful return.

However, this is not a problem that can be solved internally. The issue ultimately involves every IT department that maintains gateway or infrastructure virus protections. The generation of automated bounce-back messages is optional and can be disabled with the flip of a switch. While the intent of the automatic notifications is noble, we can safely say that it no longer provides the desired service and in fact is aggravating problems.

Unfortunately, this is a fact of today's viral world. Because spoofing is exploited by spammers, in the end, it may be the anti-spam movement that provides some relief. Such ideas as IP authentication may stem the flow of viruses into ISPs, because the spoofing will fail the authentication. This will allow ISPs to stop the messages from reaching mailboxes in the first place, reducing infections, and then stop most mass-mailings that may result. Until then, if you have any type of automatic virus notification in your email architecture, turn it off. Everyone else will appreciate it.

COMPARATIVE REVIEW

NETWARE

Matt Ham

Having set a yearly schedule for comparative testing [*see http://www.virusbtn.com/vb100/about/schedule.xml for the list of forthcoming comparative reviews - Ed*], *Virus Bulletin*'s revisit to *NetWare* this month should not come as a great surprise.

True to form, *Novell* has been busy updating its server software with yet another batch of upgrades and patches. In fact, a new patch was released on the date of finalising the test platform. However, the patch had not been uploaded to *Novell*'s website by midday GMT and therefore it was not included in this test. I suspect that a sigh of relief would have accompanied this decision as far as the submitting anti-virus developers were concerned. Even so, the patch that was used – service pack 1.1 – was a hefty 400 MB addition on top of the server installation. *Novell*'s patches have always been large, but recent patches seem to have set a disturbing trend of exponential increases in size. I await with trepidation the patch required for next year's *NetWare* review.

The line-up for this year's test was similar to that of last year's *NetWare* comparative (see *VB* August 2003, p.17), with *CAT*'s *Quick Heal* the only newcomer to the process. *NetWare* products are very much slower to be upgraded than the operating system upon which they run – leading to a general impression of them as being clunky, irritating and tending towards the user-friendliness of *NetWare 3*. Of course there are exceptions, where the products make use of the multiple methods supplied by *Novell* for integration within ConsoleOne and other admin interfaces. Some companies seem rather indecisive as to which of these paths to pursue and spread control over both. Others use Java or custom GUIs to facilitate administration from clients.

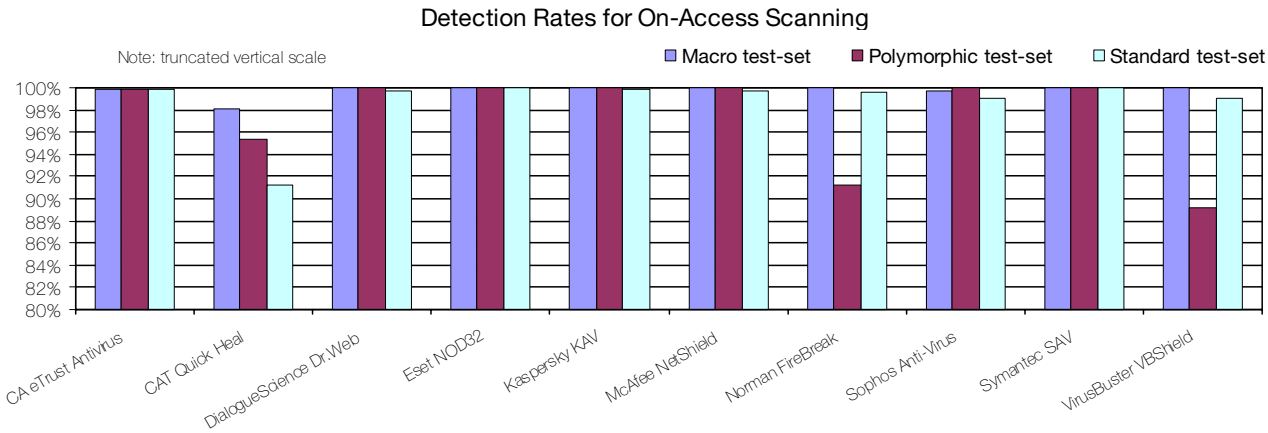
It is remarkable, given the presence of so many ways of accessing scanner functionality over so few products, that the old-fashioned methods are still the most memorable. Problems that have previously been encountered (repeatedly) for products on this platform were noted for inspection once again – it is the presence of so many recurring problems that, perhaps, explains the memories of aged interfaces behaving in unpleasant ways.

TEST SETS

The test sets were aligned with the most recent Real-Time WildList (RTWL: http://www.wildlist.org/WildList/Real-Time.htm) available two days before the submission deadline for products. Since the maintainer of the WildList was on a scheduled vacation at this time, the most recent RTWL was not particularly new. The batch of additions to the test set this month was nowhere near as gigantic as the additions made for the last comparative review (see *VB*, June 2004 p.12) and contained no new samples of any great interest. Given that the newest inclusions were all samples with well-known extensions, and the majority were worms of some sort, it was not anticipated that any of these would cause problems.

One point of note concerning testing is that both on-access and on-demand scans are performed for files located on the server. However, the accesses in the on-access tests are performed from a client machine. Thus there is an additional variable for the throughput functionality when scanning on access – namely that of data transfer from server. Since, in all cases, scanning occurs on the server, the bulk of the information transferred relates to the status of files as being infected or not – which may or may not be transferred to the client directly.

Some products provided popup warnings on the client, and others kept a real-time summary of files scanned and



On-access tests	ItW File		Macro		Polymorphic		Standard	
	Number missed	%	Number missed	%	Number missed	%	Number missed	%
CA eTrust Antivirus	0	100.00%	12	99.82%	2	99.87%	2	99.90%
CAT Quick Heal	1	99.95%	77	98.13%	882	95.37%	191	91.29%
DialogueScience Dr.Web	0	100.00%	0	100.00%	0	100.00%	3	99.69%
Eset NOD32	0	100.00%	0	100.00%	0	100.00%	0	100.00%
Kaspersky KAV	0	100.00%	0	100.00%	0	100.00%	3	99.85%
McAfee NetShield	0	100.00%	0	100.00%	0	100.00%	3	99.79%
Norman FireBreak	0	100.00%	2	99.95%	180	91.24%	11	99.63%
Sophos Anti-Virus	0	100.00%	11	99.73%	0	100.00%	17	99.09%
Symantec SAV	0	100.00%	0	100.00%	0	100.00%	0	100.00%
VirusBuster VBShield	0	100.00%	0	100.00%	602	89.12%	17	99.01%

infected. However, the level of information in these real-time views seemed to have been reduced considerably in some cases, with the result of lessening network traffic and improving scanning speeds.

CA eTrust Antivirus 7.1

ItW File	100.00%	**Macro**	99.82%
ItW File (o/a)	100.00%	**Macro (o/a)**	99.82%
Standard	99.90%	**Polymorphic**	99.87%

Computer Associates (*CA*) produces new product versions in rapid succession these days – the 7.1 version of *eTrust* coming hot on the heels of version 7.0.

The installation of *CA*'s *NetWare* product is nicely automated, following which the customary patches are applied and licence agreements accepted. The major new development with the product is that the default engine used in scanning is now the *Vet* engine rather than the *CA* engine (a fact which will explain the lack of a dedicated *Vet* product in the comparative review this month).

When initiated from the console the scanning process is very much invisible in terms of what the engine is scanning and whether infections have been found. With options set to log files only, all that results is a counter incrementing in 100-file chunks, reporting scan progress. Only when scanning has completed are infected files noted.

On-access scanning appeared to have no controls whatsoever from within the console, though it did very infrequently send messages to note infected files and it blocked access to most of the infected files offered. Scanning also seemed fairly sluggish and path selection did not offer any browsing to new scan paths or memory of past scan paths. Despite these niggles, however, detection was perfect in the In the Wild (ItW) test set and no false positives were generated in a scan of the clean test sets. The engine-swapped version of *eTrust* thus obtains a VB 100% award.

CAT Quick Heal Antivirus 7.01

ItW File	100.00%	**Macro**	98.13%
ItW File (o/a)	99.95%	**Macro (o/a)**	98.13%
Standard	91.60%	**Polymorphic**	95.37%

The most tricky part of *Quick Heal*'s operation turned out to be the installation. By default this requires access to an ADMIN user with full rights – this user being, from my memory, a throwback to some ancient version of *NetWare*. However, instead of creating an extra user it was possible to bypass the installation application and place the program files directly on the server.

Once the product had been installed scanning looked very fast indeed. The scanner itself was of the old-style console interface – such a lack of modernity had already been hinted at by the installation program, which was a DOS application.

Despite good detection rates in the wild, one missed sample in the ItW test set, together with one false positive in the clean set were sufficient to deny *Quick Heal* a VB 100% by the smallest of margins.

DialogueScience Dr.Web 4.31c

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

Dr.Web remains in the same format as for the last few reviews – that of the traditional-style NLM interface. The interface even retains its green-on-black colour scheme, which will bring back memories (fond or otherwise) to many administrators.

The point of peculiarity of this product is that on-demand scans can be instigated only as scheduled jobs – there is no provision for simply selecting an area and scanning it. *Dr.Web*'s detection rates were certainly high, meaning that it easily achieved a VB 100% award.

ESET NOD32 1.804

ItW File	100.00%	**Macro**	100.00%
ItW File (o/a)	100.00%	**Macro (o/a)**	100.00%
Standard	100.00%	**Polymorphic**	99.58%

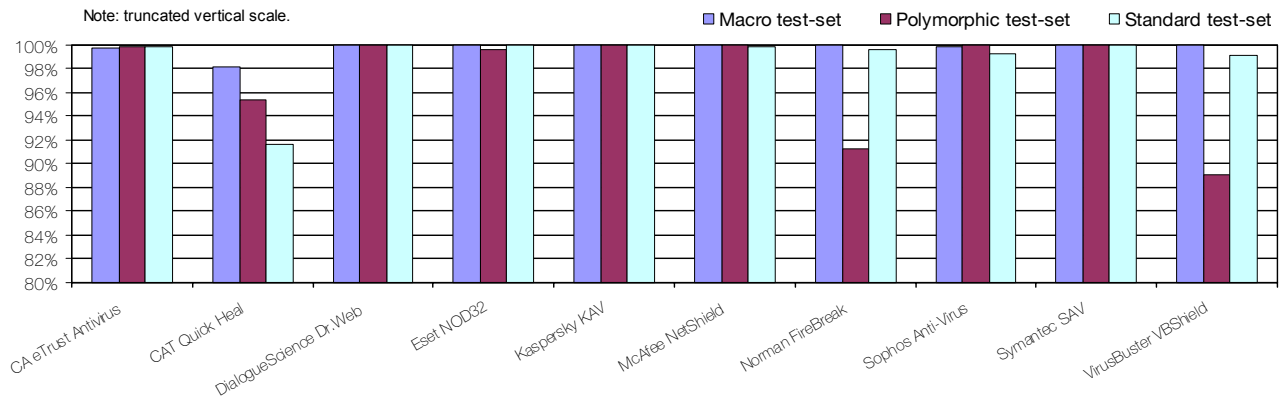
NOD32 for *NetWare* retains the distinction of functioning as two NLMs (a module each for the on-access scanner and on-demand scanner). On-demand scans are performed via a command-line interface. The product has had this configuration for as long as I can remember and, despite being archaic in nature, it serves well as a local solution on *NetWare*. However, external administration options are not the order of the day here – such would have to be created by the user. Despite the antiquated feel of the scanner, detection rates were as might be expected from *NOD32* (given its recent history in *VB* comparatives), with all samples detected both on access and on demand. Since no false positives were noted, *NOD32* is left with a new VB 100% award to add to its collection.

Kaspersky AntiVirus 5.02

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

Yet again, *Kaspersky* has tweaked and changed its method of interface – something which I found rather confusing initially. The hardest part in practice was discovering where, exactly, in the ConsoleOne view a scan is started. The process of setting tasks is carried out through ConsoleOne in a different area, which led to this

Detection Rates for On-Demand Scanning



On-demand tests	ItW File		Macro		Polymorphic		Standard	
	Number missed	%	Number missed	%	Number missed	%	Number missed	%
CA eTrust Antivirus	0	100.00%	12	99.82%	2	99.87%	2	99.90%
CAT Quick Heal	0	100.00%	77	98.13%	882	95.37%	188	91.60%
DialogueScience Dr.Web	0	100.00%	0	100.00%	0	100.00%	0	100.00%
Eset NOD32	0	100.00%	0	100.00%	133	99.58%	0	100.00%
Kaspersky KAV	0	100.00%	0	100.00%	0	100.00%	0	100.00%
McAfee NetShield	0	100.00%	0	100.00%	0	100.00%	1	99.91%
Norman FireBreak	0	100.00%	2	99.95%	180	91.24%	11	99.63%
Sophos Anti-Virus	0	100.00%	3	99.93%	0	100.00%	14	99.24%
Symantec SAV	0	100.00%	0	100.00%	0	100.00%	0	100.00%
VirusBuster VBShield	0	100.00%	0	100.00%	602	89.12%	16	99.19%

momentary confusion. However, once the two areas in which scans are controlled and initiated had been noted, the process became second nature.

Although different from its *Windows* GUI, the GUI provided in *KAV* was a welcome respite from the usual *NetWare* interface, offering a full range of configuration options and, more importantly, not necessitating the typing of long paths when scanning was required. It is of note that the GUI is the only method of scanning control here, Alt-Esc may be used to view the *KAV* console, but no interaction is possible. Scanning on access from a client was considerably faster than using the totally server-based ConsoleOne on-demand scanner. *KAV* landed itself a VB 100% award, its competence lying not merely in its ease of use.

McAfee NetShield 4.62

ItW File	100.00%	**Macro**	100.00%
ItW File (o/a)	100.00%	**Macro (o/a)**	100.00%
Standard	99.91%	**Polymorphic**	100.00%

For installation of the *NetShield* application to a server, the Java run-time environment must be available – this was a minor irritation since, by default, it was not available on the test clients. Once this has been installed the server can be loaded with the *NetWare* module. In order to interact with this, however, the console software must also be installed on the client machine.

With this completed, the scanner can be used – its look and feel is identical to that encountered with other *NetShield* products. In the past, *NetShield*'s scanning was infuriatingly slow due to excessive communication between the client and server-side portions of the server. By and large, this problem seems to have been remedied in this version of the product. Scanning is still somewhat slow on infected files, though this is a more general feature of the product over multiple platforms rather than being specific to *NetWare*.

Norman FireBreak 4.70.2282

ItW File	100.00%	**Macro**	99.95%
ItW File (o/a)	100.00%	**Macro (o/a)**	99.95%
Standard	99.63%	**Polymorphic**	91.24%

Norman offers interaction with the scanner both through

ConsoleOne snap-ins and the traditional earlier *NetWare*-style interface. The latter was used in this case, since it allows browsing to scan targets and offers no real disadvantages when compared with the more aesthetic GUI.

The same problems were encountered with this product as in last year's *NetWare* review when examining files on access: files are not necessarily scanned if opened only for reading and thus the test set was xcopied with the scanner set to purge any infected files. This worked well, though files were not purged if the read-only flag was set on them – quite an oversight. On-access scanning in this fashion was not particularly fast, even though the heuristics of Sandbox are disabled here by default. Speed is not everything of course, and with *FireBreak* showing full detection of samples In the Wild, and having generated no false positives on a scan of the clean test set, the *Norman* product achieves a VB 100% award.

Sophos Anti-Virus 3.83

ItW File	100.00%	**Macro**	99.93%
ItW File (o/a)	100.00%	**Macro (o/a)**	99.73%
Standard	99.24%	**Polymorphic**	100.00%

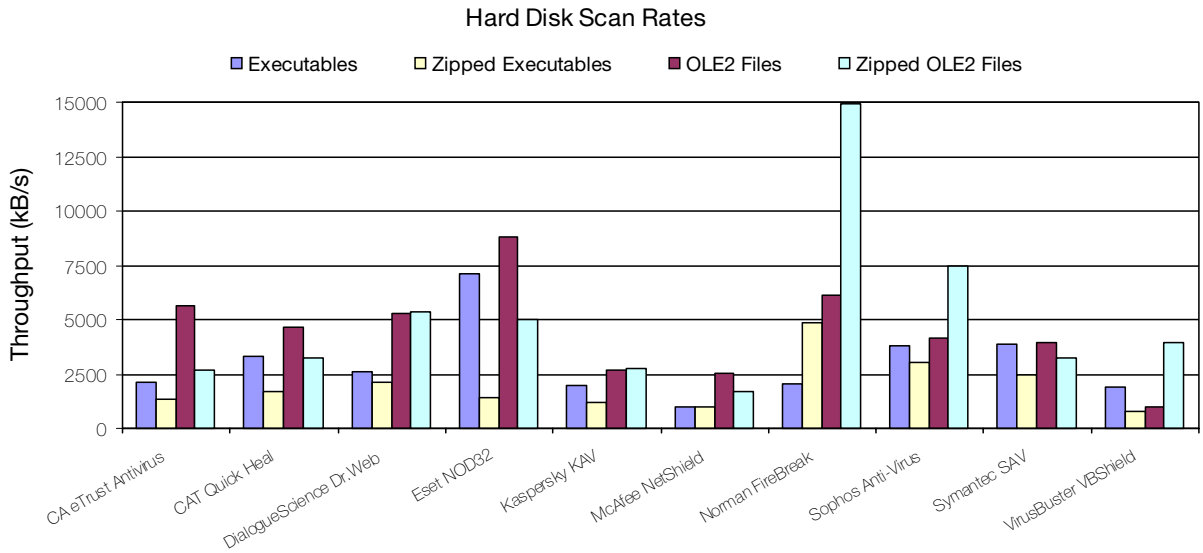
The user interface for *Sophos*'s *NetWare* product has always been something of an abomination, and this is certainly an area where four or so years might have been expected to bring improvements of some sort. My expectations were dashed, however, as the product proved about as user-friendly as a double-ended chainsaw. When scanning it is still necessary to prepend a '>' symbol to paths for recursion, it is still necessary to type each path to be scanned manually, and it is still necessary to append exact file names or wildcards to persuade the product to scan anything at all. Since paths cannot be saved without using them for scanning on every subsequent occasion, it is incredibly frustrating to scan more than one individual target.

On-access scanning is also quirky: all on-access functions are disabled by default and, when enabled, scan only files which are written to the server. Assuming that users may wish to be protected from downloading infected material from their servers, this situation can hardly be considered ideal. The one area in which *SAV* did prove that changes are being made, was detection. Several *Access* files were detected for the first time on demand, and with In the Wild detection complete, a VB 100% award is awarded to the product.

Symantec AntiVirus Corporate 9.0

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

Symantec AntiVirus (*SAV*) is another product which offers control both through a client side application and the server console. The server console controls are extremely limited, however, and the *Symantec System Center* must be installed if anything but the most basic control is to be exerted. This requirement for *SSC* in turn requires certain levels of *Internet Explorer* and *Microsoft Management Console* to be installed on an administrative



Hard Disk Scan Rate	Executables			OLE Files			Zipped Executables		Zipped OLE Files	
	Time (s)	Throughput (kB/s)	FPs [susp]	Time(s)	Throughput (kB/s)	FPs [susp]	Time (s)	Throughput (kB/s)	Time(s)	Throughput (kB/s)
CA eTrust Antivirus	260	2103.6		14	5666.7		120	1328.5	28	2664.6
CAT Quick Heal	165	3314.7	1	17	4666.7		95	1678.1	23	3243.8
DialogueScience Dr.Web	208	2629.5		15	5288.9		75	2125.6	14	5329.1
Eset NOD32	77	7103.0		9	8814.9		112	1423.4	15	4973.8
Kaspersky KAV	278	1967.4		30	2644.5		130	1226.3	27	2763.2
McAfee NetShield	550	994.4		31	2559.2		160	996.4	45	1657.9
Norman FireBreak	272	2010.8		13	6102.6		33	4830.8	5	14921.5
Sophos Anti-Virus	143	3824.7		19	4175.5		53	3007.9	10	7460.7
Symantec SAV	140	3906.7		20	3966.7		65	2452.6	23	3243.8
VirusBuster VBShield	290	1886.0		83	955.8		197	809.2	19	3926.7

machine. Since these are clearly not capabilities offered by *NetWare*, the machine cannot be the server where *NetWare* is installed. Once the rigmarole of the installation procedure was over, however, the product demonstrated few problems and the interface was the standard *Symantec* look and feel – easy enough to obtain when using a central administration tool. Also similar to other *Symantec* products, the detection rates were perfect over all sets, earning *SAV* a VB 100%.

VirusBuster VBShield 1.21

ItW File	100.00%	**Macro**	100.00%
ItW File (o/a)	100.00%	**Macro (o/a)**	100.00%
Standard	99.19%	**Polymorphic**	89.12%

On this occasion the *VirusBuster* product arrived with documentation in a PDF which was unreadable. However, this did not prove to be a major issue, since the installation was identical to that performed for last year's *NetWare* review.

The on-access functions of the product proved easy to find, though on-demand scanning took slightly longer to fathom. On-demand scans must first be assigned as a 'domain' for scanning (which is also where on-access scanning for that area is configured) then scanned as an 'option' in a different area of the interface. Where detection was concerned, matters progressed well, with a VB 100% duly awarded for the product's performance across the ItW and the clean test sets.

CONCLUSIONS

There are good and bad points to be discussed at the end of this comparative. High detection rates and an overall lack of false positives are both gratifying to observe. However, with the increased proportion of simple-to-detect worms in the ItW test set, this was not such an awesome achievement as it might have seemed in the past.

As far as reporting the iniquities of *NetWare* products is concerned, it seems that I am doomed in the same way as Sisyphus to repeat my toils forever to no avail. A brave minority have continued to add new functionality to their products in a pleasant way. However, the products which have irked me in the past through poor design or a desire to become living fossils continue to enrage me. At least I can console myself that I have not paid for such unpalatable software – though I do wonder what customers must think when presented with some of the outrageous anachronisms perpetrated by developers of *NetWare* software.

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 drive. Server running Novell NetWare 6.5 service pack 1.1. Clients running NetWare Client 4.9 service pack 2.

Virus test sets: Complete listings of the test sets used are at http://www.virusbtn.com/Comparatives/NetWare/2004/test_sets.html.

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

END NOTES & NEWS

The 13th USENIX Security Symposium will be held August 9–13, 2004, in San Diego, CA, USA. For details see http://www.usenix.org/

The National White Collar Crime Center's Economic Crime Summit takes place 17–18 August 2004 in Dallas, TX, USA. A federally-funded non-profit organization, NW3C has existed for the past 23 years to support state and local law enforcement efforts to prevent, investigate, and prosecute economic and cyber crimes. See http://www.summit.nw3c.org/.

The 19th IFIP International Information Security Conference (SEC 2004) takes place 23–26 August 2004, in Toulouse, France. Topics include intrusion detection, security architectures, security verification, multilateral security and computer forensics. For more information see http://www.laas.fr/sec2004/.

The High Technology Crime Investigation Association International Conference and Expo 2004 takes place 13–15 September 2004 in Washington, D.C., USA. The conference aims to provide training for all levels of the cyber-enforcement community from security specialists to law enforcement personnel. See http://www.htcia2004.com/.

The ISACA Network Security Conference will be held 13–15 September 2004 in Las Vegas, NV, USA and 15–17 November 2004 in Budapest, Hungary. Workshops and sessions will present the program and technical sides of information security, including risk management and policy components. Presentations will discuss the technologies, and the best practices in designing, deploying, operating and auditing them. See http://www.isaca.org/.

FINSEC 2004 will take place in London, UK on 15 and 16 September 2004, with workshops taking place on 14 and 17 September. Case studies and discussion groups will cover a range of topics including: Basel II/ IAS and IT security, prevention of online fraud and phishing scams, integrating technologies into a secure compliance framework, virus and patch management, and outsourcing IT security. For full details see http://www.mistieurope.com/.

The 14th Virus Bulletin International Conference and Exhibition, VB2004, takes place 29 September to 1 October 2004 at the Fairmont Chicago, IL, USA. For details of the conference, including online registration, conference brochure and the full conference programme (complete with abstracts for all papers and panel sessions), visit http://www.virusbtn.com/.

Compsec 2004 will take place 14–15 October 2004 in London, UK. The conference aims to address the political and practical contexts of information security, as well as analysing leading edge technical issues. For details see http://www.compsec2004.com/.

RSA Europe takes place 3–5 November 2004 in Barcelona, Spain. More information, including track sessions and speaker details are available from http://www.rsaconference.com/.

The 31st Annual Computer Security Conference and Expo will take place 8–10 November 2004 in Washington, D.C., USA. 14 tracks will cover topics including wireless, management, forensics, attacks and countermeasures, compliance and privacy and advanced technology. For details see http://www.gocsi.com/.

The 7th Association of anti-Virus Asia Researchers International conference (AVAR2004) will be held 25–26 November 2004 at the Sheraton Grande Tokyo Bay hotel in Tokyo, Japan. For details see http://www.aavar.org/.

Infosec USA will be held 7–9 December 2004 in New York, NY, USA. For details see http://www.infosecurityevent.com/.

Computer & Internet Crime 2005 will take place 24–25 January 2005 in London, UK. The conference and exhibition are dedicated solely to the problem of cyber crime and the associated threat to business, government and government agencies, public services and individuals. For more details see http://www.cic-exhibition.com/.

The 14th annual RSA Conference will be held 14–19 February 2005 at the Moscone Center in San Francisco, CA, USA. For more information see http://www.rsaconference.com/.

The sixth National Information Security Conference (NISC 6) will be held 18–20 May 2005 at the St Andrews Bay Golf Resort and Spa, Scotland. For details see http://www.nisc.org.uk/.

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

vb Spam supplement

CONTENTS

NEWS & EVENTS

VERIZON SUES SMS SPAMMERS

US mobile provider *Verizon Wireless* has filed a lawsuit against 51 individuals who, it alleges, sent over 4.7 million unsolicited commercial SMS messages to its subscribers.

Verizon claims that the defendants sent unsolicited SMS advertisements (featuring the usual roll call of products: Ephedra, mortgages, sexual enhancement pills, and so on) to its subscribers and is seeking both damages and an injunction against the defendants.

Before filing its suit *Verizon* first had to overcome the small hurdle of there currently being no US laws regarding SMS spam. Although the Federal Communications Commission was instructed by the CAN-SPAM Act to create laws to protect users against unwanted messages on mobile devices, the laws will not be in place until September 2004. As a result, *Verizon* has had to resort to filing its suit under the somewhat antiquated 1991 Telephone Consumer Protection Act (TCPA), on the grounds that the software used for SMS spamming can be considered an (illegal) 'automatic telephone dialling system'.

TWO VICTORIES, A SETTLEMENT AND A REDUCTION IN SPAM

With *Microsoft* achieving a $4 million court victory against spammer Daniel Khoshnood and his company *Pointcom Inc.*, and the New York State reaching a rather less impressive $40,000 settlement in its case against Scott Richter, aka the 'Spam King', July was a fruitful month for legal action against spammers – and not only in the US. Last month also saw the first court ruling based on the recently introduced Dutch anti-spam regulations.

Online job website *NationaleVacaturebank.nl* complained that employment broker *CVbank* had been sending job seekers' CVs to the companies advertising on its website, and giving the false impression that *CVbank* was somehow affiliated with the job website. Although Dutch anti-spam regulations only cover unsolicited email sent to individual users (spam sent to businesses is unaffected), the judge was able to rule that *CVbank* had broken anti-spam regulations because, unknowingly, it had sent unsolicited email to one of *NationaleVacaturebank.nl*'s advertisers who had given his private email address.

Meanwhile, Australian lawmakers had plenty to feel proud about when *Spamhaus* executives at the United Nation's International Telecommunications Union (ITU) meeting in Geneva revealed that they had observed a reduction in the activity of known Australian spammers since the introduction of Australia's strong anti-spam laws which came into effect in April 2004.

EVENTS

Ferris Research will present a 'Webinar' entitled "Spam Vendor Shakeouts: Who's Surviving & Who's Leading", on 18 August 2004. The relative strengths and weaknesses of the leading anti-spam vendors and service providers will be discussed. See http://www.ferris.com/.

The Organisation for Economic Cooperation and Development (OECD) will hold a workshop on spam from 8–9 September 2004 in Busan, Korea. The objectives of the workshop, which is sponsored by the Ministry of Information and Communication in Korea, are to build on the results of the Brussels Workshop on Spam, held in February 2004, and attempt to explore some of the issues and problems in greater detail. See http://www.oecd.org/.

INBOX East takes place 17–19 November 2004 in Atlanta, GA, USA. Building on the INBOX West and Email Technology Conference (ETC) held in June this year, this event will feature over 50 sessions across five tracks: Systems, Solutions, Security and Privacy, Marketing and 'The Big Picture'. See http://www.inboxevent.com/.

FEATURE

AUTOMATIC AUTHOR IDENTIFICATION FOR SPAM

Terry Sullivan
QAQD.com, USA

Informal 'eyeball' examination of spam can easily lead a casual observer to reach some fundamentally incorrect conclusions about the nature of spam. Incremental, seemingly random variations in the characteristics of spam can make junk email appear unpredictable, even volatile. Similarly, the sheer volume of spam messages makes it easy to imagine (incorrectly) that the number of unique spam sources must be very large.

Careful quantitative analysis of spam aggregates leads to exactly the opposite conclusion. Spam features are demonstrably stable over time periods spanning months, not minutes, and the weight of empirical evidence strongly suggests that the majority of spam comes from no more than a few dozen high-volume sources (see *VB Spam Supplement*, July 2004, p.S2).

Taken together, these two insights raise a tantalizing prospect. As *MessageGate*'s Kee Hinckley observed on the Anti-Spam Research Group (ASRG) discussion list [1]: "What we are sampling is not spam, but spammer targets/techniques … And it's definitely not a random population."

Given that spam features are relatively stable, and the number of unique spam sources is small, it may be possible – at least in theory – to identify systematic similarities and differences among the messages sent by various spammers. To the extent that such differences can be detected statistically, it may be possible (again, in theory) to create machine-learning technologies to serve as automatic source identification tools for junk email. Such tools would almost certainly never replace the judgment of human analysts in identifying particular sources of spam, but to the extent that they prove practical, such technologies promise a potentially dramatic reduction in total 'handling time'.

MACHINE-LEARNING

Although somewhat arcane, the broad area of statistical source identification is by no means new. Within the field of stylometry, statistically-based machine-learning technologies have already shown remarkable promise in helping to determine authorship of disputed literary works [2]. Collectively, stylometry refers to author identification methods based loosely on linguistic analysis of documents. One of the most striking results of stylometry research is the finding that non-content-bearing words, which are typically overlooked – even discarded – by traditional computational linguistic methods, are often the best discriminators among the categories of interest (authors).

The application of machine-learning technology to the problem of author identification almost always relies on 'supervised' learning methods. That is, traditional author identification efforts attempt either to determine or to verify authorship of the work(s) in question based upon comparisons between a 'training' corpus of documents and a test set. Analysis of disputed historical texts is particularly well suited to the use of supervised learning methods, because the documents being analysed are static, and a suitable training corpus is almost always available.

In the case of spam, however, the documents being analysed are stable, but not static, and training corpora are both largely unavailable and of limited utility. Thus, source identification tools for spam must necessarily be based on unsupervised learning methods, in which the similarities and differences among the categories of interest (in this case, spam source) are derived directly from the test data.

It is important to note that 'similarity' (as well as its counterpart, 'difference') is an inherently arbitrary concept, subject to equally arbitrary representation and measurement. For any given set of items, no definitive standard exists for determining their 'true' similarity, and multiple 'similarities' can be defined for any given collection of objects. Any truly robust author identification method should, therefore, be insensitive to mild-to-moderate variation in representations of inter-item similarity.

SPAMMERPRINTING

SpammerPrinting is a proof-of-concept application designed to test the viability of applying unsupervised machine-learning techniques to the task of identifying the likely authors of spam messages.

Broadly speaking, SpammerPrinting operates on any arbitrary collection of spam messages, originating from an unknown number of authors. SpammerPrinting analyses the collection of raw messages and seeks to develop a series of 'profiles' which allow messages to be grouped together based on one or more arbitrary definition(s) of 'similarity'. Messages that are grouped together by the SpammerPrinter are presumed to present a disproportionately high likelihood of originating from a single author (spammer). In this sense, SpammerPrinting simply represents a special instance of an automatic document classifier.

Authoritative reference data (a 'gold standard') is important to successful evaluation of any automatic classification technology, and the lack of such data complicates the evaluation task. In the case of spam, one example of reference data might be the historical records maintained by

Spamhaus.org as part of its ROKSO project [3], which ties spammers to spam via the domain being advertised. While ROKSO data are unlikely to be inerrant, it is even less likely that two otherwise unrelated classification processes, operating independently, would make identical mistakes in author identification. However, ROKSO data offer only sparse coverage of a limited number of high-volume spammers, making independent, externally-validated evaluation data frequently unavailable. Therefore, an alternate evaluation method – one that relies on an approximately valid surrogate – must be employed.

In the case of spammer identification, the domain being advertised presents just such a surrogate. For its undeniable limitations, URL-guessing actually presents several distinct advantages as an evaluation procedure. Foremost among these advantages is that URL-guessing permits evaluation of every prediction made, and removes the dependency on external reference data. Secondly, the criterion by which predictions are evaluated is both unique and unambiguously verifiable; each answer is either correct or incorrect. Lastly, both the predictions themselves and the evaluation results are open to public scrutiny and subject to independent verification, allowing rigorous comparative evaluation among multiple author identification methods, both automatic and manual.

There are three reasons to consider URL-guessing as an inherently stringent test of author identification techniques for spam messages. First, spammers often deliberately attempt to obfuscate the URL in an effort to bypass URL-based filtering procedures. Secondly, various messages advertising a single domain frequently bear little resemblance to each other (at least superficially). Thirdly, the number of URLs in a given data set is presumably substantially larger than the number of spammers, which serves to minimize the prior probability of accurate identification based solely on chance.

The SpammerPrinter prototype was evaluated using the domain-being-advertised as a categorical classification variable. Each message in the data set was 'held out' from the other messages, and then classified based solely on inter-profile similarity between the message being classified and the remaining messages in the test set. (This subtle point bears emphasis; the experiment did not directly compare each message to be classified against the corpus of reference documents. Rather, the classification task was based solely on comparisons among the SpammerPrinting analytical 'profiles' for the messages.)

In deference to the fact that spam messages vary systematically over time, four different collections of spam, comprising over 4,400 messages, were analysed for purposes of SpammerPrinting evaluation. The test data were obtained from multiple sources and covered a broad cross-section of spam genres and time periods from 2003 and 2004. For evaluation purposes, each of the four test collections was processed and analysed separately. However, since the classification tasks are common across all the test sets, the analytical results were pooled for reporting purposes.

Using SpammerPrinting, 45 per cent of the URL guesses were correct. For comparison purposes, just under one per cent of the guesses would be correct using random URL-guessing; a more optimal guess, based on prior statistical knowledge of the distribution of URLs among the reference spams, would be expected to yield a success rate of approximately five per cent. Thus, URL-guesses obtained from SpammerPrinting were over 45 times more accurate than random guessing, and nine times better than an empirically-based 'best guess'.

At the same time, however, the evaluation identifies two distinct limitations to SpammerPrinting. First, due to the nature of the methods used to develop the analytical profiles, high-volume spammers invariably dominate the SpammerPrinting results. By implication, SpammerPrinting does not require, nor does it benefit from, large data sets.

Secondly, SpammerPrinting analytical solutions are deliberately biased in favour of highly homogeneous groups. This deliberate conservatism leaves approximately 25 per cent of the messages in a given data set 'orphaned', assigned to 'groups' consisting only of themselves. However, to the extent that commonalities between these messages and other messages in the test set are all but absent, such information may still be helpful for prioritizing the allocation of scarce forensic resources.

These results, while preliminary, suggest that author identification based on unsupervised machine-learning technologies can potentially play an important role in the fighting of spam. Further research will determine whether, and to what extent, the methods used in SpammerPrinting might also be extended for use in creating real-time filtering technologies.

REFERENCES

[1] Posting to ASRG mailing list: http://www1.ietf.org/mail-archive/web/asrg/current/msg07534.html.

[2] Klarreich, E., 'Statistical tests are unraveling knotty literary mysteries', *Science News Online*, 20 December 2003, http://www.sciencenews.org/articles/20031220/bob8.asp.

[3] *Spamhaus.org*, 'Registry of known spam operations', http://www.spamhaus.org/rokso/.

SUMMARY

ASRG SUMMARY: JULY 2004

Helen Martin

This month postings to the ASRG mailing list concentrated largely on two issues: sender authentication and zombie emails.

Gordon Peterson raised rather a lot of hackles in his comments on sender authentication mechanisms. He claimed that spammers use fake email addresses "*as a kindness* so that complaints and bounces don't converge back on some poor victim's email inbox." He felt that, unless the spambot zombie problem is brought under control, the spam problem cannot be solved – his solution is to use a finely-grained permissions system, where each recipient authorises senders to send them "familiar and trusted types of material".

However, George Ou described Gordon's vision of such a permissions system – in which every user is expected to update their email client software – as a "grand illusion" which would not work without some form of sender authentication. He said "[the] proposal requires that all email clients [in] the world be updated, which is a pipe dream," and he pushed forward his own pet solution for combating unintentional malware execution – "*Windows XP Service Pack 2* does the best job I've ever seen of addressing this."

Another who took issue with Gordon's proposed permissions system was Andreas Saurwein, who argued that the system is flawed since users will simply authorise all senders. "People *want* to receive *anything* from their friends, family, potential friends, etc." Instead, Andreas advocates the removal of features from all mail clients: "declare MIME dead," he said. "Plain text is the only valid email format."

The bottom line, according to George Ou, is that the zombie phenomenon will be easier to manage in a post SMTP authentication world. "Instead of receiving spam from a dynamic IP address (possibly NATed as well), you will potentially receive spambot messages from a legitimate email account, [which is] a much more granular situation where you can rate limit user accounts on the SMTP server and/or shut down that email account."

Markus Stumpf felt that, if the top 500 domains comply with some form of sender authentication, spammers may stop abusing those domains, but will switch to smaller domains that have no SPF records (or 'send from all' records), thus simply shifting the problem to the small domain owners. He said "An authentication scheme based on IP addresses will be more effective, more fair and [able to be deployed] much faster than any domain name based system … anything other than blocking spam at the SMTP level is a big mistake as it shifts liability from the sender to the recipient."

Walter Dnes raised the suggestion that certain IP address ranges could be set aside for IP addresses that are not authorised to send email on port 25 to anyone other than their ISP's gateway MTA (aka 'smart host'): "Unlike other proposals, which require modifications to DNS, this idea only requires shuffling around of existing address ranges. ACLs and DNSbls already exist, and would work even better [were this idea to be implemented]."

David A. Wheeler brought up his own approach to the spam issue, which he calls "email passwords". The idea is that the user designates a word or phrase as their email password. Incoming messages with the password in their subject line can immediately be ranked as unlikely to be spam. He suggests that the password be made available to others in a shrouded way, such as in a graphic on the user's website.

Phillip Hallam-Baker asked: "why bother with the password? Just publish your email address as a gif", but David corrected him, saying that the issue was not 'how do I publish my email address on my website and not get spammed?', but rather 'how do I keep my static email address over many years and yet dump spam?'. David openly admitted that all the solutions – including email passwords – "stink", but said that, for him, the use of email passwords makes a combination of other approaches more tolerable. David's sentiments were echoed by Pete McNeil, who reported that his experience of using the same password-based approach (which he refers to as Private List [PL] codes) was very positive, and that, while not suitable for all situations, the approach works particularly well within clubs, families, church groups and the like.

Tim Bedding kicked off a discussion on the subject of zombie spam in an attempt to whittle the issue down to one clear item that can start to be addressed. John Levine was first to respond, saying that an effective way for ISPs to handle the zombie problem is to force all outgoing messages from consumers through the ISPs' outgoing servers, then rate limit the amount they send – few consumers needing to send more than 100 emails per hour. He pointed out that *Comcast* is currently monitoring outgoing direct port 25 connections and blocking any hosts that show volume spikes.

The discussion of zombie spam continued with a large number of exchanges whose topics ventured into the realms of anti-virus techniques. At the time of going to press the list members were embroiled in a long and involved discussion of the relative merits, or otherwise, of current anti-virus techniques and of blocking executable content from emails.