APRIL 2004

virus BULLETIN

The International Publication on Computer Virus Prevention, Recognition and Removal

CONTENTS

2 **COMMENT**
Year of the superworm

3 **NEWS**
New kid on the certification block
No bull?
VB2004, Chicago

3 **VIRUS PREVALENCE TABLE**

VIRUS ANALYSES
4 The wormpire strikes back
8 Doomquest: life after Mydoom

10 **LETTER**

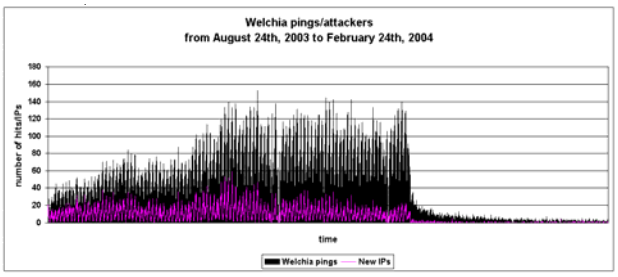
11 **FEATURE**
Environmental issues

14 **COMPARATIVE REVIEW**
Red Hat Linux 9

20 **END NOTES & NEWS**

ISSN 0956-9979

IN THIS ISSUE



THE RETURN OF WELCHIA

Firewall logs show that W32/Welchia.A's cutoff date worked well as a method of population control. But Welchia has returned. Peter Ferrie berates the author of the worm for disregarding the warning: 'a Jedi uses the Force for knowledge and defence, never for attack'.

page 4

COMPARATIVE REVIEW

Despite problems with on-access scanning and a lack of information, the overall standard of products has improved since last year's *Linux* comparative. Matt Ham has the details.

page 14

vbSpam supplement

This month: anti-spam news and events, CRM114 and DSPAM on test, ASRG summary.

'As long as the basic design of information systems is insecure our fate is to patch security with incomplete solutions.'

Berni Dwan talks with Pete Simpson and Marko Helenius

YEAR OF THE SUPERWORM

Earlier this year, *Clearswift*'s Pete Simpson predicted that 2004 would be the year of the superworm (see http://www.clearswift.com/news/PressReleases/306.aspx).

'With some trepidation, I made the prediction that 2004 will be the year of the superworm,' says Simpson. 'My definition of a "superworm" is autonomous malware that can infect all vulnerable hosts on the Internet (possibly using a new vector) and achieve its objective within a time window far too narrow for anti-virus response.

'The Sobig project set a precedent. Here we have seen a "land grab", where a significant pool of PCs (of the order of 10 million) have been hijacked for nefarious purposes. These may have included theft of banking and credit card credentials leading to identity fraud, fraudulent fronts for pornography subscriptions, anonymous spam services for sale, DDoS attacks on casinos and betting shops (protection rackets) and money laundering. The motivation of malware writers seems to have switched dramatically from an intellectual challenge to financial gain.

'It calls for little imagination to extrapolate this kind of thinking to a much more ambitious level,' continues Simpson. 'I had in mind a scenario of a worm gaining and retaining control of several million hosts. Various reports estimate that Mydoom has installed backdoors in more than one million systems. It remains to be seen whether the perpetrators planned for and are capable of exploiting this to their maximum advantage.'

Simpson feels there is a need for increased collaboration within the security world. 'We need to see a much closer collaboration between the forces of good: white hat hackers; anti-virus researchers; law enforcement bodies and ISPs. Otherwise the criminals will gain and retain the initiative.'

According to Dr Marko Helenius of the University of Tampere's Virus Research Unit, 'As long as the basic design of information systems is insecure our fate is to patch security with incomplete solutions. Currently the most vulnerable system is *Windows* because of its large usage and insufficient security. However, *Linux* and Macintosh systems are also vulnerable – it is just that *Windows* is the most likely target.'

Helenius is a man who does not like to take chances: 'Personally I do not use the *.NET* technology because I do not consider it secure enough. But then I do not use international credit cards or *Windows*-based email systems either. I suppose you could say that I am more paranoid than a typical user.

'I am concerned that the same kind of design deficiencies will be part of wireless communication,' says Helenius, and he warns that it is not only personal computers that are vulnerable: 'If security is underestimated, life-critical systems, money transaction systems and systems containing sensitive data may be put at risk.'

Who, ultimately, should be held responsible for the proliferation of superworms? 'In the end, those who create or distribute the worms deliberately are responsible,' says Helenius. 'However, I believe that part of the responsibility lies with system designers. Much of the harm comes as a result of a lack of security in information systems. Something must be wrong if a single person or a small group of people can cause the serious damage that we have seen.'

According to Pete Simpson, 'The superworm is simply a springboard to establishing a huge base of "owned" systems. It is technically feasible, the motive for financial gain is enormous, and the time is ripe.' Marko Helenius sums up: 'While predicting the future is difficult and often impossible, there is nonetheless a huge malicious potential in worm technology and what will happen is up to the criminals. I would say that we have not seen the worst yet.'

NEWS

NEW KID ON THE CERTIFICATION BLOCK

At the start of this year, *CheckVir* became the latest independent organisation to offer certification for anti-virus products, when the *CheckVir* Anti-Virus Testing project became the *CheckVir* Anti-Virus Certification program.

Like a number of other testing bodies, *CheckVir* offers two levels of certification: Standard and Advanced. The Standard certification is awarded for a product's detection capability (the product must detect all virus samples in the test set both on access and on demand), while the Advanced level examines the product's ability to repair infected objects. The results, including descriptions and summaries, are published on the *CheckVir* website. More information can be found at http://www.checkvir.com/.

NO BULL?

Recently it was brought to *VB*'s attention that affiliates of *BullGuard* have been cruising in the slipstream of *Eset*'s *NOD32* product range. Visitors to the website http://www.nod-32.com/ (note the hyphen) will find a page of promotional information about the *NOD32* product range – as one might expect. However the surprise comes when, after reading all about 'NOD 32 … the star performer of the antivirus world', the visitor tries to download the exalted product via a link at the bottom of the page. The link redirects the browser of the hapless visitor to http://www.bullguard.com/download.aspx, the download page of the official *BullGuard* website. [*According to Eset CEO Anton Zajac, BullGuard representatives have been notified and have promised to resolve the issue quickly – however www.nod-32.com remains online at the time of writing.*]

Indeed, it seems that a number of entities have seized the opportunity to freeload on the *Eset* bandwagon recently. Another duplicitous website, http://www.eset.info/, offers the *NOD32* product for sale, bills the customer, but does not provide any product. And there are reports of *NOD32* being sold on Hong Kong's black market. Ever able to spot a marketing opportunity, *Eset* CEO Anton Zajac looks on the bright side, saying, 'All these cases indicate *NOD32* is becoming popular. In fact, it is so popular, some individuals are willing to commit crime to get it.'

VB2004, CHICAGO

Online registration for VB2004 is now available on the *VB* website. The full VB2004 conference programme will be revealed later this month. See http://www.virusbtn.com/.

Prevalence Table – February 2004

Virus	Type	Incidents	Reports
Win32/Netsky	File	32451	59.74%
Win32/Dumaru	File	6843	12.60%
Win32/Bagle	File	3631	6.68%
Win32/Mimail	File	2533	4.66%
Win32/Sober	File	2438	4.49%
Win32/Swen	File	1594	2.93%
Win32/Klez	File	1160	2.14%
Win32/Sobig	File	1025	1.89%
Win32/Mydoom	File	896	1.65%
Win32/Bugbear	File	485	0.89%
Win32/Gibe	File	139	0.26%
Win32/Yaha	File	90	0.17%
Win32/Lovsan	File	74	0.14%
Win32/Funlove	File	73	0.13%
Win32/Fizzer	File	66	0.12%
Win32/Lovelorn	File	65	0.12%
Redlof	Script	58	0.11%
Win32/Nachi	File	57	0.10%
Win32/SirCam	File	55	0.10%
Win32/Magistr	File	48	0.09%
Inor	Script	47	0.09%
Win32/Parite	File	34	0.06%
Win32/Ganda	File	33	0.06%
Win32/Gaobot	File	30	0.06%
Win32/Hybris	File	29	0.05%
Fortnight	Script	25	0.05%
Win95/Spaces	File	22	0.04%
Win32/Pate	File	20	0.04%
Marker	Macro	17	0.03%
Win32/Torvil	File	17	0.03%
Win32/BadTrans	File	16	0.03%
Win32/Doomjuice	File	14	0.03%
Others		237	0.44%
Total		54,322	100%

[1]The Prevalence Table includes a total of 237 reports across 83 further viruses. Readers are reminded that a complete listing is posted at http://www.virusbtn.com/Prevalence/.

VIRUS ANALYSIS 1

THE WORMPIRE STRIKES BACK

Peter Ferrie, Frédéric Perriot
Symantec Security Response, USA

It took less than six months before W32/Welchia (see *VB*, October 2003, p.10) returned to plague us. The new version has been upgraded to attack different worms and exploit more vulnerabilities. Once again, the author of the worm intended to make a 'good' worm, disregarding the master's warning: 'A Jedi uses the Force for knowledge and defence, never for attack.'

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

After creating its mutex, the worm attempts to open a service called 'WksPatch' and query its status. If this service is set to start automatically, then the worm attempts to delete a file called 'svchost.exe' and start the service. Otherwise, the worm copies itself to the '%system%\drivers' directory as 'svchost.exe', and creates a service called 'WksPatch', using a random display name. The display name is composed of one entry from the list:

System Remote Performance License
Security Routing Network Internet

followed by one entry from the list:

Logging Procedure Event Manager Accounts

followed by one entry from the list:

Provider Messaging Sharing Client

The worm copies the existing service description from the 'MSDTC' or 'Browser' service, if available, otherwise it uses 'Network configuration manages by updating DNS names' as the description.

YOU SEEK YODA

The replication code begins by querying the *Windows* version number and the locale identification information. The worm pays particular attention to Japanese locales, which are used to activate the payload, and increase the attack range and strength.

On Japanese systems, the worm creates a file called 'temp.htm' in the current directory. This file contains dates that are politically-sensitive to the Japanese. The worm queries the '/' value in the 'HKLM\SYSTEM\CurrentControlSet\Services\W3SVC\Parameters\Virtual Roots' registry key, which returns the list of directories served by *IIS*. In the first of these directories, the worm copies the 'temp.htm' file over any file whose extension is one of: asp, htm, html, php, cgi, stm, shtm, or shtml. The worm also does this in the '%windir%\Help\iisHelp\common' directory, then deletes 'temp.htm'.

WE'RE DOOMED!

On non-Japanese systems, the worm attempts to remove W32/Mydoom.A and W32/Mydoom.B, if either of them is present. Mydoom.A is removed by deleting the 'RpcPatch' service, deleting the 'TaskMon' value from the 'Software\Microsoft\Windows\CurrentVersion\Run' registry key in both HKLM and HKCU, and deleting the 'TaskMon.exe' and 'shimgapi.dll' files from the '%system%' directory. Mydoom.B is removed by deleting the 'Explorer' value from the 'Software\Microsoft\Windows\CurrentVersion\Run' registry key in both HKLM and HKCU, and deleting the 'Explorer.exe' and 'ctfmon.dll' files from the '%system%' directory.

During Mydoom.B removal, the worm overwrites the 'hosts' file in the '%system%\drivers\etc' directory with a single default entry, but without checking the contents first. This can cause problems if the proper contents (which were replaced by Mydoom.B) have been restored already. Welchia.B also sets the 'InProcServer32' value in the 'HKCU\CLSID\{E6FB5E20-DE35-11CF-9C87-00AA005127ED}' registry key to '%SystemRoot%\System32\webcheck.dll'. This will cause a problem if the worm is run on *Windows 9x/Me*, where the correct value is '%windir%\system\webcheck.dll'.

NOT MUCH TIME

The worm checks the current date of the local machine and will remove itself if the date is after 31 May 2004, or if it is more than 120 days after the worm file was created on the local machine. Removal is performed by deleting the 'WksPatch' service, and deleting the running file using a fairly well-known routine (which is a good trick, since it is a common assumption that a running file cannot be deleted).

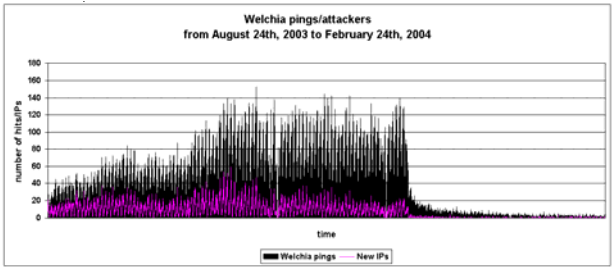


Figure 1: Daily frequencies of attacks recorded on a typical DSL machine from August 2003 to February 2004. The dramatic drop in the rate of Welchia.A ping sweeps occurred around 1 January 2004.

Welchia.A (see *VB*, October 2003 p.10) also had a cutoff date: 1 January 2004. We wondered how effective such a population control method would be. Looking back at some firewall logs, it appears to have worked very well. Figure 1 depicts the daily frequencies of attacks recorded on a typical DSL machine from August 2003 to February 2004. A dramatic drop in the rate of Welchia.A ping sweeps is clearly visible – this occurred around 1 January 2004.

IT'S NOW OR NEVER

If the worm has not expired yet, it will begin its replication phase. Generally, the worm checks for an active Internet connection, except on Japanese systems where it assumes that one exists 5 per cent of the time. The check for Internet connectivity is performed by attempting to resolve one of the names to an IP address: 'microsoft.com', 'intel.com', or 'google.com'. If a connection is not available immediately, then the worm checks again every 20 minutes.

If the locale identification information matches US-English, Korean, or Chinese PRC, then the worm checks the operating system type. If the operating system type is *Windows XP*, then the worm checks for the presence of the Messenger patch, by querying the presence of the 'SP1\KB828035' or 'SP2\KB828035' registry key in the 'HKLM\SOFTWARE\Microsoft\Updates\Windows XP' registry key. If the operating system type is not *Windows XP*, then the worm queries for the presence of the 'HKLM\SOFTWARE\Microsoft\Updates\Windows 2000\SP5\KB828749' registry key.

If the registry key(s) does not exist, the worm will wait for a random period of time, up to an hour, then silently download and install the corresponding patch. Note that since these patches require the reboot of the machine to become active, the machine remains unprotected, though it might appear to be patched correctly. (*Another problem, though independent of the worm, is that the installation of [at least the initial version of] the Messenger service patch requires that the Messenger service is running. Thus, in order to patch your potentially vulnerable system, you must first make it vulnerable by starting the service!*)

QUICKER, EASIER, MORE SEDUCTIVE

Welchia.B uses four vectors to propagate: exploits for RPC DCOM, Locator and Workstation vulnerabilities, which share a common transmission mechanism, and an exploit for WebDAV using a different transmission method.

The first three exploits inject a 'connect-back' shellcode inside the respective exploited services. Unlike Welchia.A, the shellcode does not use the socket API to connect back to the attacking machine. Instead it uses the API URLDownloadToFile() to download a copy of the worm from the attacker, and WinExec() to launch it. An advantage of using URLDownloadToFile() rather than connect() is that the IP address is embedded in the shellcode as a text string rather than as a binary. As a result, Welchia.B is no longer unable to attack IP addresses containing certain bytes.

The download location used by the shellcode is 'drivers\svchost.exe' (the worm assumes that the current directory is '%system%') but since URLDownloadToFile() first downloads files to the Temporary Internet Files directory, copies of the worm can appear in somewhat surprising locations such as: '\Document and Settings\<current user>\Local Settings\Temporary Internet Files\Content.IE5\<random directory name>\WksPatch[1].exe'.

The WebDAV exploit does not use a connect-back shellcode. It does not need to download the worm to the local machine at all, because the worm binary is encoded in the attack URL! Thus the shellcode just decodes and writes the worm binary to a file, then executes it.

From the point of view of network traffic, Welchia.B is a little more difficult to pinpoint than Welchia.A, since the pings with a peculiar payload, the TFTP transfers, and connections to port 707 are all absent, and have no equivalent in the new variant. Besides the attack ports themselves (tcp/445 for Locator and Workstation, tcp/135 for RPC DCOM, tcp/80 for WebDAV), all other ports are variable. However, the content of the packets used for fingerprinting is a giveaway.

GOT TO FIND A SAFE PORT

For URLDownloadToFile() to work, the attacking copy of the worm must open a pseudo-HTTP server on the attacking host, to which the victim host will connect. Welchia.B picks a random port for this server. It picks a first random port candidate, attempts to bind to it, and if this fails it cycles through up to 30,000 ports looking for one that can be bound successfully.

The pseudo-HTTP server of the worm accepts incoming connections and searches within the requests for the following strings in the following order: 'GET ', '/ WksPatch.exe ', 'HTTP/1.', and one blank line.

If all four strings are found, the worm sends a copy of itself to the requesting machine. However the worm does not send an exact copy. It patches three spots of its UPX-compressed binary image, corresponding to the PE header time-date stamp field, the PE header linker version field, and a region of the UPX header containing information used by UPX to decompress the file. Thus the weekend reverse-engineers will not be able to decompress the image with UPX.

However, it is possible to decompress the file by other means, and to recover the original values of the PE header fields.

In the case of WebDAV, where the pseudo-HTTP server is not used to transfer the worm, the worm image is patched by altering nybbles of the attack URL corresponding to the same spots of the PE file described above.

I AM YOUR FATHER

Following in the footsteps of its predecessor, Welchia.B uses a binary strategy, attacking nearby networks in one way, and distant ones in another. Against the local class B of the attacking host, and the class Bs above and below, Welchia.B uses the RPC DCOM and Workstation exploits. Against randomly picked class Bs, Welchia.B uses the Locator and WebDAV exploits. This is probably because the WebDAV exploit, executed against web servers, is more likely to succeed against random hosts than the other exploits using ports that are often closed at the firewall. The RPC DCOM and Workstation exploits, on the other hand, are more likely to succeed against nearby hosts that belong to the same organisation. (We wonder why the Locator exploit is used remotely, and invite anyone with a good explanation to contact us.)

The attack cycle of the worm is influenced by the locale of the attacking system. If the worm is running on a Japanese system, it will never stop attacking new machines, using 600 attack threads. Otherwise, the worm will attack machines at an average rate of one class B per three-hour period, using only 100 attack threads, and will also check its cut-off date and possibly remove itself.

Within each cycle, a single exploit is picked, and attempted against all hosts of the target network, or a pool of 64kb random hosts. The approximate probabilities for the exploit choice are as follows: 33 per cent chance for WebDAV, 33 per cent chance for Locator, 22 per cent chance for RPC DCOM, 11 per cent chance for Workstation.

When the worm uses IP address randomization, it selects a class-A network ID between 2 and 239, and random network and host identifiers within this class A. However, it avoids all non-routable IP addresses used in local networks: 192.168.x.x, 10.x.x.x and 172.16.x.x-172.32.x.x.

WEBDAV EXPLOIT

Like Welchia.A, Welchia.B attempts to exploit the targeted web server only after probing it and determining that it is running *IIS 5* and has WebDAV enabled. The exploit itself is based on the hijacking of an exception handler on the stack, leading to the execution of two stages of shellcode, and eventually the worm binary.

The same 'well-known' area of memory is referenced in the hijacked exception record that was used by W32/Blaster.A (see *VB*, September 2003 p.10) and W32/Welchia.A. Unlike Welchia.A, Welchia.B chose to use a 'jmp ebx' instruction to transfer control to the exception record. Right after the record comes a first stage shellcode, whose purpose is to decode the second stage shellcode and jump to it. The first stage shellcode is encoded in the attack URL with a series of '%u' characters (this means that Welchia.B suffers from locale dependency, just as Welchia.A did). The second stage shellcode is split into nybbles, and encoded in the URL as lower case characters, in a very similar way to the second stage shellcode of Welchia.A. In fact, the entire worm body is encoded in a similar manner, and follows the second stage shellcode in the URL. This was not the case in Welchia.A, which embedded the binary worm in the body of the SEARCH request. It is the role of the first stage shellcode to put all of these nybbles back together into a sequence of instructions. Once executed, the second stage shellcode creates the file 'svchost.exe' in the root directory of the current drive, and runs it.

The WebDAV exploit now targets random IP addresses. Welchia.B no longer uses a set of fixed class A-sized networks from which to pick its victims.

RPC DCOM EXPLOIT

The RPC DCOM exploit code is almost identical to that in W32/Blaster.A and Welchia.A. Like Welchia.A, Welchia.B targets *Windows XP* using a stack-smashing attack.

WORKSTATION SERVICE EXPLOIT

The *Windows* Workstation Service exploit in Welchia.B is twofold: one variant of it is designed to work against *Windows 2000* systems, and the other against *Windows XP*. The overall SMB transaction is very similar for both *Windows 2000* and *Windows XP*: the worm begins by fingerprinting the operating system of the target machine remotely, by sending a protocol negotiation request (on behalf of '*Windows Server 2008*') and comparing the *Windows* version to 5.0 (*Windows 2000*) or 5.1 (*Windows XP*) in the negotiation answer.

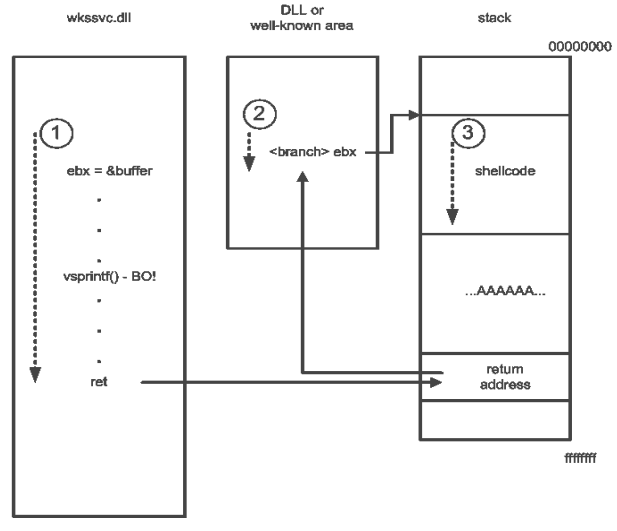
Then the worm connects to the 'wkssvc' pipe, binds to it through the service's RPC interface, and sends it a malformed request. The request is crafted differently for *Windows 2000* and *XP*. In the former case, a 'NetrValidateName2' request is issued, with a shellcode preceding an overlong ASCII name. In the latter case, a 'NetrAddAlternateComputerName' request is issued, composed of an overlong Unicode name with a shellcode embedded in it. The name is composed entirely of 'A' characters.

The control flow of the Workstation Service exploit also differs depending on the target operating system. On *Windows 2000*, the transfer of control is performed by a 'call ebx' instruction reached by overwriting a return address on the stack. At this point, the 'ebx' register points to the buffer containing the shellcode. On *Windows XP*, a classic flavor of stack-smashing is used with a hijacked return address pointing to a 'jmp esp' instruction leading to a small jumpcode directly following it in memory. The jumpcode jumps backwards to the shellcode.

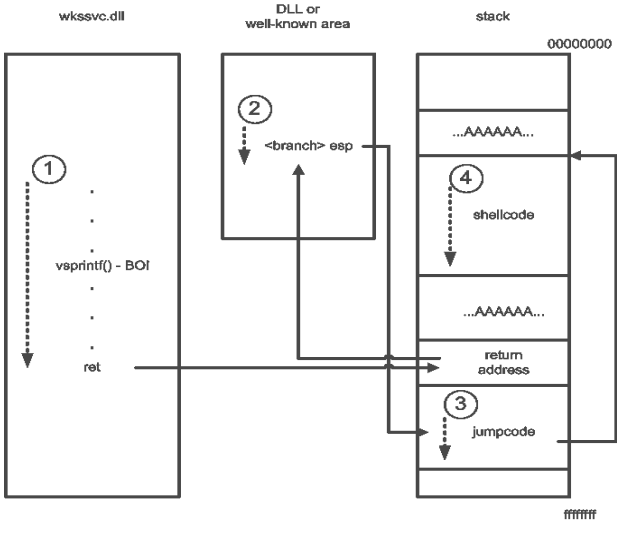
Interestingly, the addresses for the aforementioned trampoline instructions, 'call ebx' and 'jmp esp', are not always picked from a hard-coded list in the worm. Instead, they are computed dynamically on systems whose locale is unidentified by the worm, by parsing a set of DLLs for instruction patterns. If the local operating system is *Windows 2000*, the worm searches the libraries 'mprapi.dll' and 'rtutils.dll' for a 'call ebx' or a 'call esp'. If it is *Windows XP*, it searches the libraries 'ws2_32.dll' and 'wshtcpip.dll' for a 'jmp ebx' or 'push esp/ret'. Astute readers will notice that this will lead to picking the wrong trampolines 50 per cent of the time, since the *Windows 2000* exploit requires the use of 'ebx' and the *Windows XP* exploit requires the use of 'esp'. The bug is the result of a miscalculation of the index in the pattern array lookup.

The idea of looking for trampolines on the attacking host and using the obtained addresses against the attacked remote host may seem a bit disconcerting at first. The worm operates on the assumption that the target machine uses the same operating system as the attacker. This, however, is consistent with the scanning strategy used for the Workstation Service exploit, since the worm attacks only nearby class Bs.

Workstation Service exploit on Windows 2000



Workstation Service exploit on Windows XP



LOCATOR SERVICE EXPLOIT

The Locator Service exploit works almost exactly like the *Windows 2000* Workstation Service exploit described above. It smashes a stack buffer and transfers control to the shellcode through a 'jmp ebx' at a known address in memory.

THE CLONE WARS

As we write this article, the latest variant of Welchia is .D, which uses an additional propagation vector in the form of the backdoor installed by W32/Mydoom (see p.8). Welchia.D also attempts to clean systems from more competing worms. The worm war is raging!

W32/Welchia.B

Size:	12,800 bytes, UPX packed.
Aliases:	W32/Nachi.worm.b, W32/Nachi-B, Win32.Nachi.B, WORM_NACHI.B, W32/Welchi.B, Worm.Win32.Welchia.b
Type:	Exploits RPC DCOM vulnerability (MS03-026), WebDAV vulnerability (MS03-007), Workstation vulnerability (MS03-049), Locator Service vulnerability (MS03-001).
Payload:	Removes W32/Mydoom.A and W32/Mydoom.B. Patches some systems against MS03-043 and MS03-049. Defaces websites.

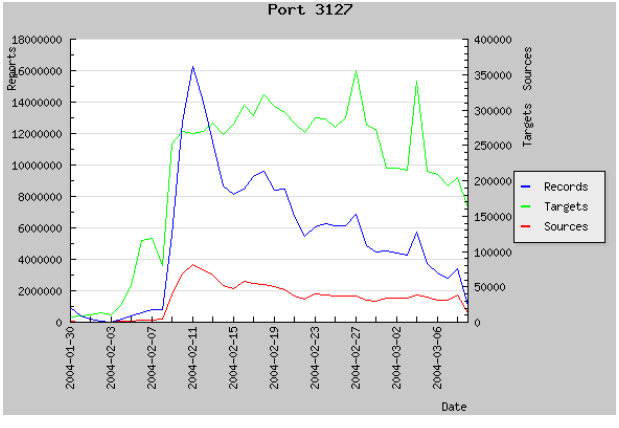
VIRUS ANALYSIS 2

DOOMQUEST: LIFE AFTER MYDOOM

Gabor Szappanos
VirusBuster, Hungary

At the end of the W32/Mydoom analysis in last month's issue of *VB* (see *VB*, March 2004 p.9) I mentioned that the large number of computers upon which the Mydoom backdoor was installed represented too tempting an opportunity to be ignored by virus writers.

Several worm traps were set up to capture malware spreading via Mydoom backdoor functionality. These have indicated a huge amount of activity in the virus writing community, as can be seen from the *Dshield* statistics:



Port 3127 activity (www.dshield.org).

There could be several reasons for the increase in the port activity:

1. Later Mydoom variants tried to replace Mydoom.A infections.
2. Other worms spreading via port 3127.
3. Malware being seeded via port 3127.
4. Sysadmins scanning for infected computers.

It is likely that a combination of these factors contributed to the huge peak, although the later Mydoom variants seem less likely to have caused the peak directly, because the two follow-up variants appeared on 28 January (too early) and 19 February (too late).

Although we had anticipated that the Mydoom backdoor functionality would not go unnoticed by the malware-writing community, the number of different malware species and the number of hits on the worm traps surpassed our expectations by a long way. Over a very limited observation period (both in time and space), more than 30 different malware samples were gathered in a month – and this is likely to be only a fragment of the complete picture.

Our worm trap statistics show that 87 per cent of the traffic comes from three worms, while the rest of the pack contributed only 13 per cent.

W32/Doomjuice.A	46%
W32/Doomjuice.B	30%
W32/Vesser.B	11%
Others:	13%

During the observation period (mid-February to early March 2004) the virus hits decreased from one in every 90 minutes to one in every 120 minutes – a considerable decrease, but the virus is still making a significant impact.

A wide range of different uses of the Mydoom backdoor functionality was observed.

1. USE OF THE MYDOOM UPDATE FUNCTIONALITY – AND MAINTAINING IT

Later versions of Mydoom (variants .E and .F) fall under this category. W32/Doomjuice.A and .B also spread via the Mydoom backdoor port and we suspect that these worms were created by the same author.

W32/Doomjuice infects computers that are already infected with Mydoom.A or Mydoom.B. It connects to port 3127, which is opened by the backdoor component of Mydoom variants. In order to bypass heuristics, the sensitive string constants (registry locations and DoS attack target) are collected from small chunks of between two and six bytes using the wsprintf formatting function.

Upon activation, Doomjuice.A checks the existence of the mutex 'zync_Z_mtx_133' to ensure that only one instance of the worm is running. Then it copies itself into the *Windows* system directory to the file 'INTRENAT.EXE'. Next, it creates a registry key under one of the following locations:

'HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run'

'HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Run'

with value '"Gremlin"' = {System}\intrenat.exe', to ensure that the virus activates automatically during each startup.

Then it creates several copies of the file 'sync-src-1.00.tbz' on each local and remote drive (enumerating drive letters c: to z:). This file contains the source code of Mydoom.A. The fact that Doomjuice.A carries the source code for Mydoom.A suggests that the two viruses may have been written by the same person. There have been various

speculations as to why Doomjuice.A carries the Mydoom source code; besides the obvious 'spread the word' exhibitionism, it is possible that the author wanted to plant the source code on several computers in order to avoid any legal consequences associated with the very successful Mydoom.A. This way the author could say that the source code found on his/her computer was the result of a virus infection.

Doomjuice.A connects to random IP addresses, looking for the backdoor installed by the Mydoom worm variants. If an appropriate host is found, it copies and executes itself there. The first octet of the IP address is selected randomly from a list (the worm avoids targeting private or unused networks), the rest of the IP address is picked randomly. Doomjuice.A does not utilize any means of spreading other than via the Mydoom backdoor. It does not replace Mydoom on the infected computer, and leaves the backdoor functionality intact. Between 9 and 12 February 2004 Doomjuice.A performed a denial of service attack against www.microsoft.com.

Doomjuice.B appeared shortly after Doomjuice.A, with the only significant difference (apart from using a different mutex and filename) being that it does not carry the Mydoom source code.

A couple of new Agobot variants (e.g. Agobot.FN and Agobot.FP) also make use of the Mydoom backdoor, but do not remove it.

2. USE OF THE MYDOOM BACKDOOR – BUT NOT MAINTAINING IT

W32/Vesser.A and .B spread via ports 3127 and 3128. They upload and execute themselves via the Mydoom backdoor, but then remove Mydoom and do not maintain the backdoor functionality.

W32/Welchia.D (aka W32/Nachi.D) spreads using known vulnerabilities (DCOM RPC, WebDAV, Workstation service) in *Windows* operating systems. Unlike earlier Welchia variants (see *VB*, October 2003, p.10 and this issue p.4), Welchia.D spreads via the Mydoom 3127 port backdoor as well.

3. MYDOOM REMOVERS

Likely to be based on information found on public mailing lists, a Mydoom-hunting self-replicating defence tool appeared recently. This is not intentionally malicious and is known as W32/Doomhunter.A.

During its operation Doomhunter displays message boxes on each step of operation, provided it was started with the DEBUG switch. When executed it copies itself into the system directory as 'worm.exe', and registers itself for startup under 'HKCU\Software\Microsoft\Windows\CurrentVersion\Run' using the key 'DELETE ME'.

Then it stops all processes corresponding to Mydoom.A and .B, and even removes the files and registry keys that are used by these two viruses. Next Doomhunter starts listening on port 3127. On incoming connections (without checking the ID dword to make sure that it is a Mydoom attempt) Doomhunter sends itself to the attacking IP.

This seems to be the only way in which this virus can get to other computers. A passive worm trap could not capture it, unless the samples came from a seeding attempt.

This sample comes as close to the (non-existent) category of 'good virus' as possible. However, Doomhunter.A has numerous deficiencies. First, it does not limit its spread, either in space (e.g. restricting itself to a local subnet) or in time (e.g. having a death time). It checks only the process names and file names; it does not check content before deleting. The fact that virus scanners detect and remove this tool will not reduce its intended actions, as (due to the nature of its responsive propagation) it will only be detected on computers that have previously been infected with Mydoom, where the infection is already removed.

While the Mydoom virus has only marginally been affected in the recent war between virus writers, several viruses do remove files and registry keys created by Mydoom variants.

W32/Welchia (aka Nachi) .B and .C remove the files and registry entries created by Mydoom.A and .B, and clear the hosts file compromised by Mydoom.B. W32/Netsky variants remove the registry entries created by Mydoom.A and .B. W32/Doomran.A removes some of the registry entries (related to webcheck.dll) that are used by Mydoom variants.

It seems that, even if a virus does not use the backdoor actively, Mydoom is sufficiently popular to be removed by viruses as a side effect.

4. SEEDED VIA THE MYDOOM BACKDOOR

The samples in this group were only seeded to infected computers using the existing Mydoom backdoors. The samples are not aware of the Mydoom backdoor, they do not use port 3127, and operate completely independently from it.

The seeding was aided by the freely available information and remote admin tools that scan for Mydoom backdoors and upload and execute files using the backdoor. One of these tools (http://studentweb.ncf.edu/rolf.rolles/scanner.zip) was published 28 January 2004 on the *TH-Research* mailing

list along with the source code. With this aid it is very easy to seed any Trojan or worm.

Several Trojan downloaders were captured in the worm traps (for example Agent.L, Apher). One of them, Shorm.A, downloads two install packages. The first, http://www.marshall911.com/~info/troj/debug.exe, is an IRC bot, while the second, http://www.marshall911.com/~info/troj/install.com, seems to be the 3.1.0 version of the free Mydoom removal tool provided by *Computer Associates*. So the Shorm.A Trojan downloader removes the Mydoom infection and backdoor, then it installs its own backdoor on the infected computer.

We also saw the backdoor RsCrt. This is an interesting piece of code. It is a very short executable (1536 bytes, uncompressed), that only opens a remote cmd.exe shell.

We captured two variants of RsCrt, the only difference between them being at the other end of the remote shell (82.192.165.54:6677 and 213.46.70.2137:10224). What is interesting is that, instead of importing KERNEL32.DLL functions, RsCrt finds its load address via Process Environment Block -> ProcessModuleInfo->Initorder ([[[[FS:30h]+0Ch]+1Ch]+8]), and browses the exported function names, comparing it to the built-in checksums. This is nothing new, just something that is rarely seen in backdoors.

END NOTE

Life after Mydoom has been very interesting. Based on different reports, between 500,000 and one million computers were infected with Mydoom, all of which were wide open to intruders using the backdoor. This defined a subset for the virus scene, and several different viruses fought to win control of this subset – in a situation reminiscent of the old core wars era.

Over the past month the authors of all of the major new virus families felt they had to refer to Mydoom one way or another – it seems this virus made a big impact on virus writing as well.

Some of the worms maintained the backdoor, some removed the infection entirely, others just replaced it or replicated via it. The constant decrease observed in port 3127 activity indicates that the new viruses are consuming this subset, because all except the later variants of Mydoom decrease (or leave unchanged) the number of affected computers.

This means that this type of port activity will diminish slowly. The reason this will happen slowly is because the most active Mydoom hunters are still the Doomjuice variants, which maintain the backdoor functionality.

LETTER

ON MYDOOM (RE)NAMING

Reflecting on the virus description in the March 2004 issue of *Virus Bulletin* (see *VB*, March 2004, p.9), the name 'Mydoom' is so infectious that Gabor Szappanos's article does not even list aliases for the worm. In my opinion AV companies should show more restraint when choosing names for malware. 'Mydoom' is probably the worst ever abuse of responsible virus naming!

'Mydoom' is a first-person virus name, suggesting that each computer user is being targeted selectively. It conjures up violent scenes from *Doom*, the famed shoot-em-up computer game by *ID Software Corp*. Trying to make a splash is legitimate, creating hysteria is not.

In the beginning there were several names for this particular piece of malware. I encountered five: Novarg, Shimgapi, Worm.SCO, Mimail.R, Mydoom.

Mimail.R was a bad name, as deeper analysis found no code related to the prolific administrative mass-mailer.

Worm.SCO was a bad name, because it could have harmed the reputation of said company (*SCO*), possibly resulting in litigation. Furthermore, there is no guarantee that future variants of the worm will not attack linux.org, www.ibm.com or the website of any other organisation. (A similar problem was observed with the Hungarian-specific 'Magold' worm [see *VB*, August 2003 p.4], whose later variants had no connection with Maya Gold, the local porn movie celebrity. In hindsight, the 'Auric' alias used by some AV vendors was a superior naming choice.)

Shimgapi or Novarg would have been good names, but these certainly would not have found their way to front pages of newspapers or dominated *CNN*'s top-of-the-hour coverage.

Consequently, 'Mydoom' (the name) started to spread, replacing all the other naming variants rapidly. Is the AV community suffering from the same moral insanity that has affected some politicians' labelling logic? Which AV vendor will be the first to rename 'rusty.old.two-stroke.truck.damaged' to '45-minute.ready.WMD.dropper'?

Enforcement of a uniform, real-time virus-naming convention may help root out media-frenzy naming habits, as well as curbing the proliferation of virus names (Beagle.B has *six* other aliases). It should be agreed upon and put online ASAP.

Tamas Feher, 2F 2000, Hungary

[*In next month's VB, Andrew Lee looks at the hunt for a universal naming convention, examining the usual reasons why a universal convention is sought, and why those are not necessarily the reasons one is needed.*]

FEATURE

ENVIRONMENTAL ISSUES

Dr Igor Muttik and Daniel Wolff
McAfee AVERT, Network Associates, UK

You may have read the article 'System Disinfection' in the June 2002 issue of *Virus Bulletin* (see *VB*, June 2002, p.10). The article raised the issue of complete system restoration after an infection. Cleaning, however, is a secondary action after detection – and in order to obtain the right clues about malware one sometimes has to gather a lot of evidence first. Let's discuss how such detective work can affect different aspects of detection, reporting, avoidance of false alarms, performance and cleaning.

DETECTION

Traditionally, anti-virus software has been wholly file-centric. Detection was always based on the content of 'a file' (plus, of course, a few boot sector viruses, file-system infectors like Dir-II and oddities like the '3APA3A' virus). However, with today's interconnected networked environment and the abundance of 'system' infectors we have to look at a broader picture. Examining the environment in which 'a file' resides can shed a great deal of light on what it is up to, as well as provide lots of heuristic clues.

In some circumstances the gathering of comprehensive environmental information is necessary in order to distinguish an innocent program from a security breach. For example, the same mIRC client software may be used in a legitimate way (installed in the 'Program Files' folder under the filename MIRC32.EXE) or it could be a component of the IRC/Flood 'bot' [2] (hiding, UPX-packed, under the name SHELL32.EXE in the %TEMP% folder). Detection of the file in the first example would constitute a false alarm. Missing the file in the second would be a mistake.

Another example is legitimate FTP server software. On the one hand this may be running on a system with the user's consent. However, it is a different matter altogether if that same FTP server was launched by a BAT file hiding it with the 'Hidewindow' utility (thus running a file-sharing server without the user's knowledge). Indications of whether we have the first or the second scenario can be gathered from the Registry and folders' structure.

MALWARE PACKAGES

These days we come across malicious software packages quite frequently.

A good example of such a bundle is the W32/Tzet worm (which bears certain similarities to many other IRC/Flood packages). A description of this threat [3] reads rather like a cooking recipe because so many different components are used ("take several potentially unwanted applications: 'HideWindow', 'PSKill' and 'RemoteProcessLaunch', sprinkle with a Trojanized mIRC client and mix well with several scripts and BAT files").

The functionality of the W32/Tzet worm comes together only when the relationships between all the components are considered. The BAT component is a tiny dictionary-based password cracker for weak network shares; 'RemoteProcessLaunch' starts a remote copy on a weakly-protected remote share; 'HideWindow' hides the backdoor part and the Internet communication function is provided by mIRC.

This is what we call the 'bundling' correlation. It is a very useful 'environmental' heuristic factor.

FILE HISTORY

The history of a file bears important information for any heuristic score. For example, if an executable file was received by email or downloaded through a 'Kazaa' P2P file-sharing network it should be treated with greater suspicion than if it was copied from a CD-ROM during an install.

An even more radical approach to file history would be to take into account the source of HTTP, FTP or POP3/IMAP downloads during gateway or firewall scanning. It is not unreasonable to treat binaries from, say, a personal web page with a higher level of suspicion than those from the website of a reputable software company.

Furthermore, the ability to check the validity of a digital signature on a binary (or even a PE checksum field in the file header) can also come in very handy.

EMULATION

The problem with detecting changes in the environment is that they are not visible before the malware has run. This makes it more difficult to ensure proactive and heuristic detection of threats in the context of mail and other perimeter scanning (e.g. FTP or HTTP). The same would apply to a situation before malware had activated on a workstation.

We cannot examine such useful indicators as the Registry, memory and disk contents because the threat has not activated yet. All that is left for static analysis is the 'bundling' correlation and, possibly, the filenames. The

alternative, of course, is to perform dynamic analysis and examine the behaviour of a program when it is executing under an emulator. This, however, can be an extremely time-consuming process.

One possible solution could be a combination of the two approaches. For example, emulating only the main installation BAT script that puts executables under predefined names into the folders.

In essence, we are talking about broadening the heuristic approach from a per-file basis to multiple files. This will also mean a switch from emulating the behaviour of a single program to the emulation of a whole computer in a network.

HEURISTICS

It is one thing when an email arrives with an executable attachment, but it would be significantly more suspicious if that email had an HTML body incorporating an exploit. Such a correlation may raise the level of heuristic suspicion significantly. Another heuristic factor could be if the same email was detected by an anti-spam filter (or correlator) – worms are more and more frequently being distributed in huge spam batches.

A mismatch between filename and file type is a strong indicator of the presence of malware – an EXE file with PIF extension is an obvious example. Another heuristic rule could be a mismatch between attachment type and attachment contents (for example, 'audio/x-wav' type for an executable file).

Detective work becomes even more important when trying to improve generic and heuristic detection. For example, if you know that a program has been communicating (or trying to communicate) through port 6667 (IRC) or 3127 (W32/Mydoom), this is an additional factor to flag as suspicious.

The opening of network ports for listening, for example, can indicate a backdoor – although it can, of course, be legitimate. If, additionally, the program tries to hide under the name of a system service (like SVCHOST/IEXPLORE/LSASS/SPOOLSV/SERVICES), this can only increase the 'suspicion factor'.

PACKING

Another important 'environmental' factor is packing. If a BAT file is hidden inside BAT2EXE, that is curious. If a Win32 PE file is packed inside UPX, that is a bit dodgy. But when a program is packed with Morphine on top of 10 layers of AsPack, this is extremely suspicious.

For products that have the ability to 'see through' the packing it becomes quite important to embed the 'packing factor' into the heuristic score. Additionally, PE packers can be obfuscated using primitive patching. This is another strong indicator of foul play.

To summarise the issues regarding malware detection we suggest the following 'environmental' indicators can be used:

- Filenames, extensions and type.
- File locations.
- Registry contents.
- 'Bundle' correlation, inter-file relationships (frequently embodied in installation scripts).
- Packing and obfuscation.
- Mismatching content.
- Malware correlation.
- File history.
- Ports opened.
- Outgoing traffic on non-standard ports.

REPORTING

Another important reason for evaluating the environment is to provide more useful reporting. There is a world of difference, for example, between the W32/Mydoom virus detected on a computer in an incoming email message and the same virus detected in an active state running in memory.

The first is a minor nuisance (and not unexpected, if your name is Dan [4]), which would likely only put a satisfied smile on your face owing to the efficiency of your mail AV solution. The second, however, is a serious security breach and requires a re-image of the computer to guarantee safety – this virus provides a backdoor through which anything could have been installed or modified while the PC was exposed.

Another area where extensive reporting may help stop malware would be the blocking of potentially dangerous attachments in the perimeter products. Many companies block all executable content in emails – but some do not, because they need them.

AV products could report packed PE files properly and granularly (and, perhaps, other similar mildly suspicious content like double extensions or archived executables), thus letting the users adjust the filtering threshold to suit their needs. For example, allowing only plain EXEs (non-packed and with proper extensions) in emails would

greatly reduce the risk of infection while still letting the business flow.

FALSE ALARMS AND PERFORMANCE

Taking into account 'environmental' issues during cleaning may help achieve better performance results and avoid false alarms.

Let's imagine a virus, V, drops components A, B and C when executed. Let's say B is a backdoor, while A and C are files that are not worth detecting separately at all (for example pure text files, GIF images or MP3 files).

It is common for AV companies to include files V, A, B and C in their collections. These get into the hands of the anti-virus testing bodies too and become part of evaluation suites, putting pressure on AV companies to detect all of these files. An example is the W32/Parrot virus, which drops an MP3 file.

Recently we had a report that one AV scanner was detecting a version of MIRC32.EXE as a Trojan. This could, of course, have been because the scanner in question does not have a 'potentially unwanted' category yet. However, a more likely explanation is that the detection of this file was introduced to perform better cleaning of a multi-component threat.

Most scanners cannot perform actions on files that have not been detected – this is an obvious technological limitation. And if not for the pressure of this old 'per-file' detection/cleaning legacy this file may not have had to be detected at all.

In the example of virus V dropping files A, B and C, the best strategy may be to detect only the malicious components (virus V and backdoor B), while removing files A and C during the cleanup operation. This still ensures full protection, while providing increased performance and reduced risk of false alarms, and helps to achieve perfect cleaning.

An unexpected facet of cleaning arises with what *McAfee* calls 'potentially unwanted applications'. These are legitimate programs about which users are frequently warned (e.g. adware programs such as *Gator* [5], which comes bundled with other programs that usually disclose the fact that they are ad-supported). The unexpected part comes from the fact that software that the user wants may stop functioning if *Gator* is removed.

Even if there is no direct dependency of other programs, there could be a legal aspect of removing the 'unwanted' *Gator* program in a situation where its presence is dictated by the conditions of licensing the 'wanted' part of the software package.

Obviously any decent product must be able to take into account such dependencies before attempting to remove the adware component. Observed commonality of adware makes this issue one of the most important for the consumer AV market.

NOT INCLUDING DETECTION

There is another example of tricky cleaning affecting the detection. Careful readers may have noticed that, traditionally, *McAfee VirusScan* misses three files in the *Virus Bulletin* standard test set in comparative reviews (see, for example *VB*, February 2003, p.20). These three files are W32/Nimda.A files with TMP extensions.

For performance reasons *McAfee* products do not scan temporary files by default (this speeds up *Microsoft Office* operations and significantly reduces the installation time for several setup packages). At the same time, TMP files do not pose any threat, as they cannot be executed accidentally. We made the conscious decision not to include detection of these stray files and only to remove them as part of the cleanup operation.

CONCLUSION

We believe that the sooner anti-virus developers depart from the legacy of the old 'per-file' concept and make the necessary modifications in their products, the better. Doing more detective work in assessing the environment of any scanned object will improve proactive detection rates, improve speed, reduce false alarm rates and enhance cleaning.

There are really no drawbacks to this approach except the effort of a bit of research – which, for virus researchers, should be fun. (Although, unfortunately for Matt Ham that would make life even more complicated while testing products for *VB* reviews!)

REFERENCES

[1] Jong Purisima, Vincent Tiu 'System Disinfection', *Virus Bulletin*, June 2002, p.10.

[2] IRC/Flood.bq: http://vil.nai.com/vil/content/v_100181.htm.

[3] W32/Tzet: http://vil.nai.com/vil/content/v_100519.htm.

[4] W32/Mydoom.A: http://vil.nai.com/vil/content/v_100983.htm.

[5] Adware-Gator: http://vil.nai.com/vil/content/v_100854.htm.

COMPARATIVE REVIEW

RED HAT LINUX 9

Matt Ham

Since *Virus Bulletin*'s last *Linux* comparative review (see *VB*, May 2003, p.18), the *Linux* bandwagon has rolled along steadily, gaining momentum and providing ever more financial reason for vendors to provide a product for the platform. Only 11 products were submitted for last year's *Linux* review, two of which do not reappear this time (*GeCAD* having been absorbed by *Microsoft* and *Norman* not having submitted on this occasion) – however there are five newcomers: *SOFTWIN*, *NAI*, *CAT*, *Grisoft* and *Eset*.

Only two VB 100% awards were achieved in the 2003 *Linux* comparative, with the on-access components of the products proving the largest source of trouble. A year later, it is clear that the feasibility of various scanning methods has been tested in the marketplace, and there is an appearance of greater homogeneity in the methods used. 'Appearance' is the key word, however, because the scanning methods used in several products are not mentioned in any detail, leaving only guesswork to determine how scanning is performed. Seasoned campaigners would, at this point, berate me with cries of 'RTFM!' – which would be easier if manuals were always provided, but more of that later.

The primary methods for on-access scanning on *Samba* shares (this being the chosen area for the tests) can be divided into those in which the scanning is performed only on the *Samba* share, and those in which all files are scanned. Where only the *Samba* share is scanned, the predominant method is the insertion of 'vfs object = <filename.so>' into the smb.conf file. This can be applied globally or on a per-share basis. Where scanning of all file accesses is desired a kernel object may be inserted and scanning performed by means of a daemon. The most popular way of doing this is via the *Dazuko* module.

Some problems occurred in on-access scanning. The problems with the vfs object method of scanning seemed primarily due to overloading of the *Samba* processes. This resulted in slowed scanning, the creation of large numbers of *Samba* processes and permanent or temporary termination of the *Samba* connection. *VirusBuster*'s developers warned that there were known, temporary problems with their product where 10,000 infected file accesses were exceeded. Other products demonstrated similar problems without prior warning.

More problems resulted from the old chestnut of insufficient information. In some cases documentation was lacking and in other cases it was hidden. For many products the final destination of the installed files was a mystery, which made finding and activating on-access scanning unneccesarily difficult. Simple tasks, such as loading daemons, may seem obvious to a developer, but for a user who is not even sure how on-access scanning is intended to operate, the absence of instructions for such tasks is infuriating.

DAZUKO

Available from http://www.dazuko.org/, *Dazuko* is an open source file access control interface, designed to be used over a full range of *Linux* and *BSD* platforms. While it is linked with *H+BEDV*, which has provided much of the funding for the project, *Dazuko* demonstrates sufficient independence for other companies to have felt no qualms about using its functionality. The total package is less than 60 KB in size, so distribution is easy from a logistical point of view.

Since such low-level interaction with the file system is not possible without direct reference to the kernel on the current machine, *Dazuko* must be compiled locally before it can be used. It was with a degree of trepidation that I noted in the readme for the module that the instructions were for a 'quick and dirty' installation. From past experience these words can be translated as 'this won't work, refer to the 1000-page manual for a better way'. On this occasion, however, the quick and dirty method proved simply to be quick. All that was required was to allow the module to configure before making it, inserting it and activating it – each process being a matter of one command line which, for the truly lazy, can be cut and pasted from the readme. In all, *Dazuko* was a pleasure to work with.

Where *Dazuko* was concerned, the low-level nature of the scanning initiated by the module was something of a problem when interacting with on-demand scanners. By choice, the on-access components of products are disabled whenever on-demand scans are carried out during comparative tests. However, sanity-checking exercises on single files demonstrated that there were cases where on-demand scanning would show no detections, since the on-access portion of the scanner was denying access to infected objects. This was not a major problem during testing, though in real-world situations this could be more of an issue.

THE TESTS

In general the testing methodology varied little from the standard methods used for the *Windows* tests. *Samba* testing was an exception, since it is unique to these *Linux* tests. The test client runs *Windows NT4 SP 6* and is configured to access the collection of infected files in addition to various directories used in file transfers. These transfers are for the installation of the applications on the *Linux* machines and extraction of results – during testing of the on-access

components there is no other file activity between the two machines. Detection is considered to be confirmed when file access through fopen() is denied to an infected file. In cases where this does not trigger detection, denial of file copying is considered to be equivalent. For products which are unstable over large test sets the *Samba* process was restarted between tests.

Alwil Avast! 0.2.0

ItW File	100.00%	**Macro**	99.56%
ItW File (o/a)	100.00%	**Standard**	99.36%
Linux	70.00%	**Polymorphic**	93.58%

One of the two products which gained a VB 100% award in the last *Linux* comparative, *Avast!* is the first of those in this review that use the *Dazuko* module. The installation method is via shell scripts and is slightly long-winded as a result. This consists of installing *Dazuko*, installing the core *Avast!* engine modules, installing the scanner modules and finally activating the scanning daemon manually. Thankfully, the documentation was thorough.

On-demand scanning ran without any problems at all. However, there were some issues with on-access scanning when the whole test set was passed through in one batch. This caused a scattering of unblocked files distributed randomly across the set. With a slightly slower throughput, however, the detection became consistent and approached that of the on-demand scans. Historically, such cases have resulted in a VB 100% award, along with the caveat that, in this version at least, some files may be missed. It should be noted, however, that this version of the product is not fully released as yet, so some problems would be expected.

CAT QuickHeal X Gen Ver 7.01

ItW File	100.00%	**Macro**	97.51%
ItW File (o/a)	100.00%	**Standard**	83.42%
Linux	40.00%	**Polymorphic**	91.84%

Another *Dazuko*-based scanner, *QuickHeal* also uses shell scripts to perform installation. However, there was some initial confusion in the installation procedure, since the installation shell script was not tagged as an executable file. This was easy to correct, if mystifying, and once this obstacle had been overcome the process was completed quickly. With full detection of viruses in the In the Wild (ItW) test set and no false positives, *QuickHeal* gains a VB 100% award.

DialogueScience Dr.Web 4.31.1

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

Dr.Web consists of two RPM packages: one for the command line version and another for the *Samba* functionality. The scanning of *Samba* accesses is performed via the insertion of a vfs object reference in the smb.conf file, thus offering on-access detection only for *Samba* accesses. *Dr.Web* flagged 12 files as suspicious, but no full-blown false positives. The product's detection rate was much more impressive, with all infected files detected. *Dr.Web* thus overcomes its recent blip in performance and adds another VB 100% to its current collection.

Eset NOD32 2.01 1.650

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

The installation of *NOD32* involves two RPM files and an additional shell script, on top of the required *Dazuko* compilation. As far as on-demand scanning was concerned matters were simple enough, with only one miss in the whole test set. This was of W32/Lovelorn.A in its DLL form – a new entry in the standard test set and not a worrying miss. Matters were not so trouble-free, however, where on-access scanning was concerned. The documentation provided was copious in quantity, but lacking any form of troubleshooting information where scanning failed to initialise. The developers were consulted, and the problem investigated further – happily for *Eset* the result was a VB 100% award.

FRISK F-Prot Antivirus 4.3.5 3.14.8

ItW File	100.00%	**Macro**	100.00%
ItW File (o/a)	100.00%	**Standard**	99.82%
Linux	66.67%	**Polymorphic**	99.91%

Having been reviewed as a standalone product recently (see *VB*, December 2003, p.14), the installation of *F-Prot Antivirus* has become something of a routine. The insertion of the preload instructions into the *Samba* configuration file must be performed manually, but it is documented well enough for this to be only a minor chore.

On-access tests	ItW file		Macro		Polymorphic		Standard		Linux	
	Number missed	%	Number missed	%	Number missed	%	Number missed	%	Number missed	%
Alwil Avast!	0	100.00%	18	99.56%	112	93.58%	18	99.42%	9	80.00%
CAT QuickHeal	0	100.00%	102	97.51%	1277	91.84%	315	83.30%	26	40.00%
DialogueScience Dr.Web	0	100.00%	0	100.00%	0	100.00%	0	100.00%	0	100.00%
Eset NOD32	0	100.00%	0	100.00%	0	100.00%	1	99.91%	0	100.00%
FRISK F-Prot Antivirus	0	100.00%	0	100.00%	2	99.91%	4	99.69%	1	93.33%
F-Secure Anti-Virus	0	100.00%	0	100.00%	0	100.00%	0	100.00%	0	100.00%
Grisoft AVG	0	100.00%	12	99.71%	425	83.72%	46	97.11%	16	48.33%
H+BEDV AntiVir	0	100.00%	56	99.26%	522	87.18%	34	98.42%	4	85.33%
Kaspersky Virus Scanner	0	100.00%	0	100.00%	0	100.00%	0	100.00%	0	100.00%
NAI VirusScan	N/A	-	N/A	-	N/A	-	N/A	-	N/A	-
SOFTWIN BitDefender	4	99.12%	21	99.49%	11	97.46%	69	97.49%	6	83.33%
Sophos SWEEP	N/A	-	N/A	-	N/A	-	N/A	-	N/A	-
Trend ServerProtect	0	100.00%	0	100.00%	215	95.77%	11	99.56%	4	93.33%
VirusBuster VirusBuster	0	100.00%	0	100.00%	102	91.45%	11	99.60%	39	13.33%

On-access the scanning performed well, though there was some noticeable slowing especially on some of the polymorphic test sets. On demand there was no such slow down, leaving one to conclude that the issue lies with the on-access component. Despite a slow performance in places, the product's detection rates were certainly sufficient to qualify for a VB 100% award.

F-Secure Anti-Virus 4.60 3100

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

Being related to *F-Prot Antivirus* by way of the *FRISK* engine within, the performance of *F-Secure Anti-Virus* was expected to be similar. The installation method marked the first difference between the products, in *F-Secure*'s case consisting of a shell script which leaves little configuration to the administrator. Similarities in scanning performance existed to a certain degree, in that the on-access engine showed distinct slowness on certain polymorphic files. However, the slow speed of scanning did not affect the product's thoroughness and *F-Secure Anti-Virus* earned a VB 100% easily.

Grisoft AVG 7.03 262

ItW File	100.00%	**Macro**	99.63%
ItW File (o/a)	100.00%	**Standard**	97.36%
Linux	81.67%	**Polymorphic**	83.72%

Returning to *Dazuko*-powered scanners, *AVG* was of mixed pleasure to install. Installation is via an RPM file which distributes files to various directories scattered across the file system. Upon detecting these it was necessary to activate an update application and to install a licence key, again through an application. This over-complicated matters to an

irritating degree. After installation, however, the *AVG* scanners performed well, and there were no further problems of any sort. No false positives were registered, and In the Wild detection was exemplary. *Grisoft* thus receives a VB 100% award.

H+BEDV AntiVir 2.1.0-9 6-24-0-39

ItW File	100.00%	**Macro**	99.55%
ItW File (o/a)	100.00%	**Standard**	98.45%
Linux	57.00%	**Polymorphic**	87.18%

As might be expected from the company's support of the *Dazuko* development process, *AntiVir* uses the *Dazuko* engine as its method of scanning. The installation procedure consists of a fairly lengthy shell script, similar to that found in other products. The installation and operation of *AntiVir* was without any noticeable problems as far as functionality or stability were concerned. It is thus of little surprise that *H+BEDV* is in receipt of a VB 100% award.

Kaspersky Virus Scanner 5.0.1.0/#1

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

Kaspersky's product was another of those whose documentation caused problems. Initial installation is via RPM file, after which two perl scripts must be run – these were discovered more by luck than judgement. The vfs object was duly added to the smb.conf file, though at this point the *Samba* share simply ceased functioning. Activation of the daemon scanner solved this, though it was difficult to find mention of this workaround in the documentation. The product's detection rates were faultless, however, and *Kaspersky* earns a VB 100% award.

NAI VirusScan 4.32.0 4333

ItW File	100.00%	**Macro**	100.00%
ItW File (o/a)	N/A	**Standard**	99.79%
Linux	80.00%	**Polymorphic**	100.00%

The first product in this review not to offer an on-access component, *VirusScan* arrived as Tar.Z files which were inaccessible to a standard *Red Hat* installation. This did not bode well, though the installation, through a shell script, continued smoothly after this point. With no on-access functionality, a VB 100% award for *VirusScan* was always an impossibility, though on all other fronts the performance was close to admirable. Misses which did occur were limited to archived or packaged objects, since *VirusScan* does not handle archives in its default installation state.

SOFTWIN BitDefender Console 7.0(2489)

ItW File	99.12%	**Macro**	99.49%
ItW File (o/a)	99.12%	**Standard**	97.55%
Linux	60.00%	**Polymorphic**	97.46%

BitDefender opts for an RPM format for installation, though this is packaged within a RUN file so as to offer both licence and configuration functionality. This seemed to be a good compromise between convenience and information. Despite this configuration functionality, however, paths must be inserted manually. The scanning functionality is provided by a vfs object reference in the SMB.conf file. One peculiarity was noted, in that the extension listing for scanned files appeared to be set so that only files with lower-case extensions were scanned. By default, the entire *VB* test set is fully upper case. This obstacle was overcome quickly, but certainly warrants a mention.

Scanning on access proved more of a problem. Files were missed both on access and on demand, and the connection to the *Samba* share had a tendency to break after 10,000 files passed through the scanner. These problems have been acknowledged by the developers and should be rectified in the future.

Sophos SWEEP 3.79

ItW File	100.00%	**Macro**	99.80%
ItW File (o/a)	N/A	**Standard**	99.60%
Linux	60.00%	**Polymorphic**	100.00%

The second product to be tested with no on-access component, *SWEEP* is designed without any such functionality. *SWEEP* is installed though a shell script and is accompanied by documentation in the form of a helpful readme. With no on-access component, *SWEEP* is not eligible for a VB 100% award, though detection rates for on-demand scanning were of the same high standard as seen from *Sophos* in recent *Windows* comparative tests.

Trend ServerProtect 0403Nov03D021004

ItW File	100.00%	**Macro**	100.00%
ItW File (o/a)	100.00%	**Standard**	99.72%
Linux	93.33%	**Polymorphic**	95.77%

On-demand tests	ItW file		Macro		Polymorphic		Standard		Linux	
	Number missed	%	Number missed	%	Number missed	%	Number missed	%	Number missed	%
AIwil Avast!	0	100.00%	18	99.56%	112	93.58%	18	99.36%	11	70.00%
CAT QuickHeal	0	100.00%	102	97.51%	1277	91.84%	313	83.42%	26	40.00%
DialogueScience Dr.Web	0	100.00%	0	100.00%	0	100.00%	0	100.00%	0	100.00%
Eset NOD32	0	100.00%	0	100.00%	0	100.00%	1	99.91%	0	100.00%
FRISK F-Prot Antivirus	0	100.00%	0	100.00%	2	99.91%	2	99.82%	6	66.67%
F-Secure Anti-Virus	0	100.00%	0	100.00%	0	100.00%	0	100.00%	0	100.00%
Grisoft AVG	0	100.00%	15	99.63%	425	83.72%	42	97.36%	10	81.67%
H+BEDV AntiVir	0	100.00%	28	99.55%	522	87.18%	34	98.45%	9	57.00%
Kaspersky Virus Scanner	0	100.00%	0	100.00%	0	100.00%	0	100.00%	0	100.00%
NAI VirusScan	0	100.00%	0	100.00%	0	100.00%	3	99.79%	3	80.00%
SOFTWIN BitDefender	4	99.12%	21	99.49%	11	97.46%	70	97.55%	10	60.00%
Sophos SWEEP	0	100.00%	8	99.80%	0	100.00%	5	99.60%	7	60.00%
Trend ServerProtect	0	100.00%	0	100.00%	215	95.77%	9	99.72%	4	93.33%
VirusBuster VirusBuster	0	100.00%	0	100.00%	102	91.45%	11	99.66%	39	13.33%

ServerProtect is the only product to offer a GUI in the *Linux* comparatives. It is supplied as a BIN file, which acts as a wrapper for an RPM, giving licensing details. The GUI aspect of the software is reached by use of an http connection, performed from a local or remote browser. There was a slight problem in that a stray " ' " was added to one URL when triggering the GUI, though once this had been removed (manually) there were no further problems in program operation.

In terms of detection, *ServerProtect* performed very well, gaining a VB 100% award. Considering the addition of a GUI to the program the throughput rates on the clean sets were not noticeably slower than the bulk of other products.

VirusBuster VirusBuster 1.12.019

ItW File	100.00%	**Macro**	100.00%
ItW File (o/a)	100.00%	**Standard**	99.66%
Linux	13.33%	**Polymorphic**	91.45%

VirusBuster is one of those products where the exact nature of the scanning is not mentioned in the process of installation – this being through the faceless method of an RPM package. The developers contacted me after submission, having discovered that there were issues with the *Samba* scanning functionality, which had a tendency to break after 10,000 files. In fact, problems were encountered sooner than this and the testing of on-access file interception was performed in small batches through the use of blocked copy operations. With these limitations in mind, the product's detection rate was very good and a VB 100% award is duly gained for detection. According to the developers the problems noted in scanning are no longer present in shipping products.

CONCLUSION

By the nature of the complex interactions required, on-access problems are at least understandable, if not forgivable. Testing puts unique strains on a scanning engine,

Hard Disk Scan Rate	Executables			OLE Files			Zipped Executables		Zipped OLE Files		Linux Files	
	Time (s)	Throughput (MB/s)	FPs [susp]	Time (s)	Throughput (MB/s)	FPs [susp]	Time (s)	Throughput (MB/s)	Time (s)	Throughput (MB/s)	Time (s)	Throughput (MB/s)
Alwil Avast!	31	17643.0		8	10438.7		21	7591.3	13	5739.0	3	10007.6
CAT QuickHeal	50	10938.6		13	6102.6		34	4688.7	13	5739.0	2	12866.9
DialogueScience Dr.Web	179	3055.5	[12]	11	7212.2		83	1920.7	14	5329.1	4	6755.1
Eset NOD32	33	16573.7		4	22666.8		21	7591.3	4	18651.9	2	16887.9
FRISK F-Prot Antivirus	81	6752.2		3	23333.5		43	3707.4	5	15226.0	2	13510.3
F-Secure Anti-Virus	119	4596.1		13	6102.6		89	1791.2	31	2406.7	6	4503.4
Grisoft AVG	77	7103.0		9	8718.0		52	3065.7	12	6217.3	10	2702.1
H+BEDV AntiVir	108	5064.2		5	16527.9		33	4830.8	6	12863.4	6	4740.4
Kaspersky Virus Scanner	148	3695.5		13	6102.6		67	2379.4	19	3926.7	7	4032.9
NAI VirusScan	77	7103.0		9	9015.2		60	2656.9	13	5739.0	5	5749.1
SOFTWIN BitDefender	586	933.3	[1]	6	12395.9		26	6131.4	7	10812.7	6	4289.0
Sophos SWEEP	56	9766.6		11	7212.2		35	4554.8	12	6217.3	4	6433.5
Trend ServerProtect	77	7103.0		6	13222.3		34	4688.7	7	10658.2	5	5404.1
VirusBuster VirusBuster	200	2734.7		7	11173.8		119	1339.6	13	5739.0	6	4579.8

which do not relate to any real-world situation likely to be encountered by users. Take, for example, the case where a *Samba* share automatically disconnects when many viruses have been detected. As part of a test scenario this may cause upset, but as part of a network where viruses must be contained, this behaviour may perform a useful function.

Informational problems, on the other hand, cause me far more grief and are multi-part in nature. Many products for this test were packaged using the RPM format which, due to its monolithic nature, does not allow for very much in the way of obvious documentation. Some developers packaged the RPM within a tarball, with a readme file as the sole other object present – this was very welcome.

It would be useful to know the location of the files which are installed. With there being no consensus as to the correct place for anti-virus software to be installed, the impression arrived at after this test was that all products wish to be unique in this respect. Installing to root, /local, /opt, /etc, /usr/lib, /usr/local/lib and many other locations gives a first-time user very little idea of where to locate their new scanner. Particularly irksome were those products which scattered components over four or more directories.

In addition, the basic command line to activate the scanner is rather a handy piece of information, especially if paths are not set up. On several occasions I searched for appropriate-sounding filenames in desperation, having no other clue as to how to initiate the scanner. This is particularly frustrating where daemons are required to be activated manually and are not mentioned at any stage in the documentation (if useful documentation exists at all).

On a more positive front, the overall standard of products has improved since last year, which is reflected in the number of VB 100% awards gained. As product lines become more stable it is hoped that the level of documentation will also show improvement.

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 running *Red Hat Linux 9*, kernel build 2.4.20-8 and *Samba* version 2.2.7a. An additional machine running *Windows NT 4 SP 6* was used to perform read operations on the Samba shared files during on-access testing.

Virus test sets: Complete listings of the test sets used can be found at http://www.virusbtn.com/Comparatives/Linux/2004/test_sets.html.

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

END NOTES & NEWS

The 3rd Annual DallasCon Wireless Security Conference takes place 1–2 May 2004, in Dallas, TX, USA. The conference will feature two tracks: one dedicated to the latest trends and threats in wireless security and a second focusing on general information security. For details see http://www.dallascon.com/.

The EICAR Conference 2004 will be held in Luxembourg City, from 1–4 May 2004. EICAR 2004 will feature only one stream, which will give in-depth coverage of issues including malware, critical infrastructure protection, legal and operational issues, and identity management and social issues. More information is available from http://www.eicar.org/.

The 2004 World Computer and Internet Law Congress takes place on 6 and 7 May 2004 in Washington D.C., USA. The event, presented by the Computer Law Association, will focus on providing practical advice on current IT law. For full details see http://www.cla.org/.

The VII Ibero American Seminar of Security in Computer Science Technology will be held 10–15 May in Havana, Cuba. Topics will include: security on servers and network servers, database security, computer forensics, cryptography, intrusion detection, authentication and control mechanisms, policies and standards of security, ethics and legal aspects of computer security. For more information see http://www.informaticahabana.com/.

The Black Hat Briefings and Training Europe takes place 17–20 May 2004 in Amsterdam, The Netherlands. For more information see http://www.blackhat.com/.

RSA Japan takes place 31 May to 1 June 2004 at the Akasaka Prince Hotel, Tokyo. For details see http://www.rsaconference.com/.

The Sixth Annual International Techno-Security Conference will be held 6–9 June 2004 in Myrtle Beach, CA, USA. Topics will include computer forensics, Homeland security, intrusion detection, 'street smarts for cybercops', technical counter-terrorism, privacy issues, and security policies. For full details see http://www.technosecurity.com/.

The 10th Annual Gartner IT Security Summit takes place 7–9 June 2004 in Washington, D.C., USA. Topics include critical infrastructure protection, securing the workplace, security software and security strategies. See http://www3.gartner.com/.

NetSec will take place 14–16 June 2004 in San Francisco, CA, USA. The conference programme covers a broad array of topics, from the management issues of awareness, privacy and policy to more technical issues like wireless security, VPNs and Internet security. For full details see http://www.gocsi.com.

Internet Security & Payments takes place 15–17 June 2004 in London as part of the Internet World UK event. For details see http://www.internetworld.com/.

MIS Training will host a CISO Executive Summit in Geneva on 16 and 17 June 2004. This event for IT security leaders will cover the unique issues faced by CISOs. For more information contact Yvonne Hynes on +44 20 77798975 or email yhynes@misti.com.

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. A track will be dedicated to 'Security and Control of IT in Society'. For information see http://www.laas.fr/sec2004/ .

The 14th Virus Bulletin International Conference and Exhibition, VB2004, takes place 29 September to 1 October 2004 at the Fairmont Chicago, IL, USA. The conference programme, including abstracts for all papers, will be available to view online from mid-April. For more information about the conference, including online registration and details of sponsorship and exhibition opportunities, see http://www.virusbtn.com/.

The 31st Annual Computer Security Conference and Expo will take place from 8–10 November 2004 at the Marriott Wardman Park in Washington, D.C., USA. More details will be available in due course from http://www.gocsi.com/.

SUBSCRIPTION RATES

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

Editorial enquiries, subscription enquiries, orders and payments:

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

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

vb Spam supplement

CONTENTS

S1 **NEWS & EVENTS**

S2 **PRODUCT REVIEW**
CRM114 and DSPAM

S4 **SUMMARY**
ASRG summary: March 2004

NEWS & EVENTS

A BIT OF R&R

Some choose Yoga, others choose a glass of wine and a soak in a hot bath, and some, apparently, choose reading spam as their preferred method of unwinding from the stresses of the working day.

It was with some disbelief that *VB* read, in the *Wall Street Journal* (http://www.wsj.com/), about a 45-year-old New Yorker who likes to receive spam. The behaviour described seemed (to *VB*) so unthinkable that the piece read more like satire than an article in a respected news journal – according to the *WSJ* the gentleman in question spends hundreds of dollars a week making spam-related purchases, claiming that spam helps him 'lose the stress of the day' and even goes as far as to say that 'good spam ... leaves [him] feeling blessed'. With a recent survey by anti-spam company *MailShell* finding that eight per cent of respondents had bought products via spam, one wonders whether the problem is being approached from the wrong angle. Perhaps, rather than concentrating solely on those who send spam, attention should be turned to those who perpetuate the problem by *responding* to it …

NEW SENTENCING, NEW LEGISLATION

The United States Sentencing Commission (USSC) – the body tasked with refining the sentencing portions of new legislation – met last month to review public comment on how to enforce the CAN-SPAM Act.

The initial suggestion was to use the guidelines for fraud sentencing as a model for CAN-SPAM. However, both the National Association of Criminal Defense Lawyers and the Electronic Frontier Foundation objected to this idea, protesting that violations of CAN-SPAM 'do not inflict nearly the same level of harm … or constitute the same seriousness [as] fraud.' Another proposal is to implement a sliding scale of sentencing based on the number of emails sent, with a maximum of three years for the most prolific spammers. The USSC is expected to vote later this month to send its CAN-SPAM amendments to Congress, with the amendments expected to take effect (unless Congress acts to change them) from 1 November 2004 .

Elsewhere, the government of Singapore is considering introducing legislation to deter spammers operating within the country; Australia's spam laws come into effect this month, and the German government (which was criticized last year by the European Commission for not implementing spam restrictions in accordance with an EU directive) is also set to introduce anti-spam laws this month.

SHRINKING VIOLETS

A report released by *Gartner* last month predicts consolidation in the anti-spam industry and a rapid contraction of the pool of anti-spam vendors over the course of this year. The report suggests that, by the end of 2004, the current tally of about 40 vendors in the anti-spam market will have shrunk to fewer than 10. Maureen Caplan Grey, one of the authors of the report, likened the expected change to that seen in the anti-virus market a few years ago. Her recommendation was that enterprises pick a solution from a company which treats anti-spam as a piece of the overall email security picture.

EVENTS

The Institute for Spam and Internet Public Policy's International Spam Law & Policies Conference takes place 30 July 2004 in San Francisco, CA, USA. Details can be found at http://www.isipp.com/events.php.

The first Conference on Email and Anti-Spam (CEAS) will be held 30 July to 1 August 2004 in Mountain View, CA, USA. Further details can be found at http://www.ceas.cc/.

PRODUCT REVIEW

CRM114 AND DSPAM

Pete Sergeant

The concept of using automated statistic-based categorization to identify spam is not all that recent, but the idea really came into the public eye with the publication of Paul Graham's 'A plan for spam' in August 2002. Although the first papers on the subject were presented in 1998, Paul Graham was the first to publish his success using the idea.

Graham used what he called 'Bayesian filtering' to identify spam – in essence, he trained a spam filter to give individual words a score based on the frequency of their occurrence in spam and non-spam email. He then tried to deduce if a given email was spam by looking at the scores of the words within it. Graham claimed some very impressive results: 99.5 per cent detection of spam, with no false positives.

A number of open-source anti-spam products sprung up almost immediately after the publication of Graham's article, which implemented his 'Naive Bayes' algorithm for catching spam.

Graham later published another article refining his methodology. Clearly this aroused some interest in automatic classification – today there are numerous anti-spam products that do some form of statistical classification.

Obviously not everyone is taken with this approach for identifying spam. The method works best when each individual user has their own 'probability dictionary' – which tends to be considered impractical by commercial anti-spam vendors, mainly due to the fact that a good dictionary will be around 4 to 10MB, per user.

This article serves as a reflection on installing and testing the effectiveness of two open-source statistical classifiers which claim a lower error rate than humans in detecting spam. I hope that this article will also serve to open the debate on anti-spam testing best practices.

THE CORPUS

The corpus is a snapshot of my incoming personal mail over a two-week period, and sits at 6,500 messages, almost exactly equally divided between spam and 'ham'.

The emails were sorted into spam and ham by hand, and then by running various anti-spam products over them – if any disagreements arose between the products and my initial judgment, the message in question was rechecked. Therefore, while there are probably one or two misclassified messages in the corpus, I consider it likely that the number of erroneous messages is tiny – and probably comparable with the amount of human error in training that would occur in a real-world situation.

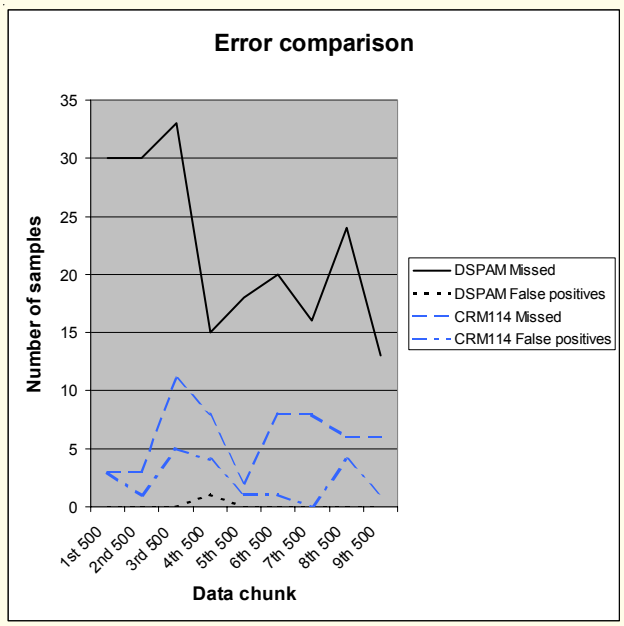
Deciding what constitutes spam and what does not is a debate that is beyond the scope of this article – I consider spam to be email that I do not wish to receive, and my corpus is marked accordingly. Therefore, legitimate email bounces are considered ham; bounces that originate because someone spoofed my email address to send spam are classified as spam.

For obvious reasons of privacy, I am not able to share this corpus publicly. I do consider it a difficult corpus however: a lot of email is spoofed from my domain, and I receive emails sent to any address at my domain – therefore I receive a lot of bounces. Also, I am subscribed to a number of mailing lists with differing levels of spam protection – a product relying on mailing list headers could well be tripped up by the fact that the 'OpenBSD misc' mailing list is 99 per cent ham, and 1 per cent spam, thus making any short spam sent to me via that mailing list look a lot more 'hammy' than might otherwise be the case.

PRODUCTS UNDER REVIEW

I chose to test *DSPAM* and *CRM114* after having read a recent *Slashdot* article (http://slashdot.org/) which stated that the author of *CRM114* had claimed that both *CRM114* and *DSPAM* had a better detection rate than humans. A bold claim, and one I felt compelled to investigate.

While *DSPAM* (see http://www.nuclearelephant.com/projects/dspam/) is intended purely as a spam identifier, *CRM114* (see http://crm114.sourceforge.net/) is a more general classifier – however, it has very specific instructions



	DSPAM				CRM114			
	Correct	Missed	FP	Accuracy %	Correct	Missed	FP	Accuracy %
1st 500	470	30	0	94.0	494	3	3	98.8
2nd 500	470	30	0	94.0	496	3	1	99.2
3rd 500	467	33	0	93.4	484	11	5	96.8
4th 500	484	15	1	96.8	488	8	4	97.6
5th 500	482	18	0	96.4	497	2	1	99.4
6th 500	480	20	0	96.0	491	8	1	98.2
7th 500	484	16	0	96.8	492	8	0	98.4
8th 500	476	24	0	95.2	490	6	4	98.0
9th 500	487	13	0	97.4	493	6	1	98.6
Overall	4300	199	1	95.6	4425	55	20	98.3

on how to set it up as a spam classifier. Additionally, it asks the user for hints, such as the user's email address, to help it with identification. As far as I could see, *DSPAM* does not do this.

I found both products easy to install – however, I am an experienced UNIX system administrator; those who do not fit that description may have a little more trouble, but the documentation provided with each product is clear.

METHOD

I felt that, to be practical for the average user, a categorizer should be producing fairly good results after being trained on 1,000 pieces of spam, and 1,000 pieces of ham.

A testing harness was written for both products – messages were taken out of the database, and the product asked to classify them. If the product got the classification wrong, it was informed, meaning that the product's accuracy should have improved after each mistake.

No data is given about the speed of scanning – the overhead of the Perl-based testing harness would render these results meaningless – however, both products comfortably classified at least three messages per second on an old *Pentium II* machine.

Both products were trained on errors, and the 'testing' phase was started after the 2000th message. As accuracy was expected to increase over the course of testing, a figure is given for the products' accuracy in every chunk of 500 messages, as well as the overall accuracy during the testing phase.

RESULTS

The results can be seen in the table above, with a graphical comparison of the accuracy of the two products shown opposite.

Neither product performed quite as well as I had hoped – however, both were impressive. While, at first glance, it looks like *CRM114* has a slight advantage over *DSPAM*, it is worth noting the very small number of false positives identified by *DSPAM* in comparison to the number identified by *CRM114* – in general, users would rather see a couple of pieces of spam in their inbox than know that they have to comb through their quarantined messages for potentially important email.

In practice, my incoming email is filtered through a tool I have written to filter out spam bounces – however these were left in the corpus, marked as spam. About half of the misclassified email fell into the 'bounces' category – were I to implement either product on my system for use in day-to-day filtering, I imagine the accuracy would be higher than is shown here. Also, spam bounces of this type are a problem for only a small number of users.

The fact that *DSPAM* generated only one false positive is very impressive indeed – and it is also the reason why I would choose *DSPAM* over *CRM114* in a production environment.

SUMMARY

ASRG SUMMARY: MARCH 2004

Pete Sergeant

The highlights of this month's ASRG activity include postings on mail flow, biological analogies, accreditation databases and S/MIME.

Dave Crocker posted a link to Patrik Fältström's mail flow control chart, showing the different paths emails can take. This allows anti-spam researchers to examine potential solutions against which mail 'flows' they might break, as well as helping identify which flows should be broken – email going through open relays, for example. The chart is at http://www.ripe.net/ripe/meetings/ripe-47/mailflows.pdf.

Eugene Crosser wondered whether it would be interesting to consider the 'email system' as a biological organism and look upon spammers as parasites. Yakov Shafranovich pointed out that similar analogies have been drawn between real-world viruses and computer viruses/worms and wondered whether these analogies could be extended to spam. Kurt Magnusson was opposed to the idea, saying that the main difference between real-world viruses and spam is that viruses exist to 'self-propagate', while spam is dependent on the 'work-order' of its creator, and its existence is driven purely by economic returns.

Yakov pointed the list to *Microsoft*'s Caller-ID paper. Bob Atkinson indicated that this was just one piece of the puzzle – those interested in *Microsoft*'s anti-spam measures should point their browsers to http://www.microsoft.com/spam/.

Brett Watson noted that he has seen a number of failed spam attempts sent to rather odd addresses recently, and traced this back to poor address scraping on the part of spammers. Notable examples were '3dtfbw@...' which he assumed was a mis-scrape of '=tfbq@...' in quoted-printable format, and 'asrg@...' where the correct address is 'famous-asrg@...'. He surmised that if you want to reduce spam to your address, you should simply add a non-alphanumeric character to the user part!

One of the most interesting threads this month kicked off concerning ISIPP's accreditation database. According to ISIPP, the IADB is 'a judgement-neutral list, asserting that those listed therein meet certain standards and/or are personally known to ISIPP to be responsible mailers'.

The main upset about the database was the degree to which the ISIPP maintained the right to make arbitrary judgements on who was included on the list, rather than including automatically any host who met a certain set of criteria. It was pointed out that this allows the IADB to block spammers looking for loopholes without facing legal action.

Seth Breidbart was quick to defend the system, saying that, just like blacklists, it is up to users who they trust to help them make judgements on the 'spaminess' of an email – if one doesn't like the IADB's policies, there is no obligation to use it.

Anne Mitchell, President and CEO of ISIPP, clarified that a listing in the database is not an endorsement, more a certification that a certain set of criteria have been met, as well as a way of determining that a sender is who they say they are – and stressed that one shouldn't automatically whitelist on it. Yakov still had some reservations: 'Many blacklists,' he said, 'say that you should not use them as the only source of information, but many ISPs ignore that and do it anyway.'

The return codes given by the IADB were listed and discussed (http://www.isipp.com/iadbquery.php), which sparked a discussion on standardization of DNS-based blacklist/whitelist/accreditation return codes. Anne Mitchell pointed out that the codes used by the IADB were a subset of Craig Hughes' standard code set (which can be found at http://www.isipp.com/codelist.php) and that these are not merely proposed codes – the IADB is already being used 'to help ISPs and spam filters make … decisions for more than 30 million email boxes'.

A posting about fake (postal) letters informing recipients that a former sexual partner had tested positive for HIV sparked a thread on authenticated email. Barry Shein suggested that a good idea would be some form of 'higher-cost authenticated messaging using something like SSL', which could be used when the recipient absolutely had to know that an email had come from a given sender.

John Levine reported that, at a recent meeting of the Anti-phishing Working Group, the suggestion had been made that a certification authority could be set up, to issue certificates to banks which could translate into a sign in the user's browser that the site they are visiting is a genuine financial institution. ('It's not from your bank unless it has a Golden Dollar Sign in the corner.')

Alan DeKok pointed out that a weakness with the current authentication system is that a name is essentially meaningless – one has no way of knowing if a given company's legitimate address is 'example.com' or 'example-company.com' – and suggested that structured host naming might be a solution.

The topic moved on to S/MIME, and the question was raised: 'How does the need to buy a certificate and demonstrate that you are allowed to use it help combat spam?' Doug Royer answered in some detail, explaining that it made it very easy to trace and block given spam, and presumably the extra overhead needed would push spam into non-profitability.