NOVEMBER 2003

virus BULLETIN

The International Publication on Computer Virus Prevention, Recognition and Removal

CONTENTS

ISSN 0956-9979

IN THIS ISSUE

TORONTO TOPS FOR VB2003

A Mountie, a magician, a magnificent setting and two days packed with discussion, debate and delegates. All this and not a hurricane in sight …

page 8

MAP READING

During the last week of August 2003 there was temporary relief from the tedium and monotony of the life of a virus analyst: a virus for a new platform had appeared. Mikhail Pavlyushchik describes the near excitement of analysing the first virus to infect *MapInfo* files.

page 4

AT YOUR SERVICE

Matt Ham cracks his knuckles and puts 21 anti-virus products for *Windows 2003 Server* to the test.

page 13

Nov 2003 VIRUS BULLETIN 100% www.virusbtn.com

vbSpam supplement

Included in this month's *VB*: the inaugural issue of *VB Spam Supplement* – bringing you news, views and developments from the world of anti-spam.

'Maybe one day we will see an issue of Virus Bulletin *presenting anti-virus for OS XIV.'*

Jakub Kaminski
VB Technical Editor 1995–2003

SO LONG, FAREWELL, AUF WIEDERSEHEN, ADIEU …

[After eight years with Virus Bulletin, Jakub Kaminski has decided the time has come to hang up his Technical Editor's hat. Jakub's knowledge, patience, expertise and friendship have been invaluable to those of us tasked with producing the magazine over the years, and the current team will be sorry to see him go – but we are pleased he will be able to enjoy a little more spare time at the end of each month. VB is delighted to welcome Morton Swimmer on board as Technical Editor and we look forward to a long working relationship with him. Ed.]

There are goodbyes resulting from fundamental disagreements. This is not one of them. There are goodbyes that follow endless bouts of fiery arguments. This is not one of them. There are goodbyes that seem as if they are the natural progression and reflection of the passing time. That's the one!

When I was asked to become the technical editor of *Virus Bulletin* eight years ago, things happened so quickly I didn't have a chance to think the decision through. My first issue of *VB* was also the second issue for Ian Whalley as the chief Editor (June 1995 – also the second month for Megan Skinner as Assistant Editor). Either the situation was desperate, or Ian felt like gambling (he and I had never met) because, while I was still at the stage of enquiring about my potential duties in the role, Ian had taken it as read that I was in and expected me to do my part the very next day (almost).

Before I had a chance to verify whether I could swim I was in at the deep end and it was too late to panic. It took only a couple of months for me to realise what Frisk (whose shoes I was attempting to fill) had meant when he mentioned the time requirements and in particular the 'PC Virus Update' section of the magazine. Over the next four years, creating summary descriptions and selecting signature patterns for new viruses became one of the most tedious tasks for which I have ever volunteered. Oh boy, was I happy when the virus templates section of the magazine was phased out!

I've been lucky to see *Virus Bulletin* grow from strength to strength, and I've been lucky to see the *Virus Bulletin* conference develop into an annual event not to be missed. However, one thing is more important than anything else: I've been lucky to meet fantastic people and see them in action. I have watched Ian, Nick, Ceskie and Helen working their proverbials off in order to deliver another high quality issue of *Virus Bulletin* on time every month. I've seen Megan, Alie, Fraser, Matt, Bernadette and Pete take on their part of the stress in the process of giving birth to a new issue of your favourite magazine, 12 times a year; regular as clockwork. And despite all that pressure, I have been lucky in that, no matter how hectic things were in Abingdon, I knew that all I could count on was big a smile. Finding friendships that extend past the working duties was an unexpected bonus. Thank you guys!

There have been a few changes in the magazine that I did not expect to witness during my shift. The new logo was certainly a surprise, mainly because the old one had been with us for as long as I can remember. Following that, the new full colour graphics seemed like a natural progression. I also saw the very first comparative review of anti-virus products for *Linux*. It's almost like an official recognition that the number of non-*Windows* users, non-*Windows* malware and anti-virus offerings for *Linux* has reached a level that can no longer be ignored. Who knows, maybe one day we will see an issue of *Virus Bulletin* presenting anti-virus for OS XIV.

Eight years ago it seemed like I was one of very few people available or willing to put on the technical editor's hat – today, I would have to face sizeable competition. The number of industry names willing to add yet another task to their busy lives was astounding. No doubt this is a reflection on the reputation *Virus Bulletin* has built over the years. Passing the job on to Morton makes me very calm about the future. Good luck!

NEWS

ROGUES' GALLERY





Photographs of all the goings on at VB2003 are now available for browsing on the *VB* website. If you haven't seen yourself in the photos here or on p.8, you can browse online by category, by photographer or search for yourself and your favourite AV personalities by name. *VB* would like to thank all reportage photographers who have allowed us to include their photographs in the montage. If you see any errors in or omissions from the captions, or have photographs that you would like to contribute, please email pete.sergeant@virusbtn.com. Whether it's to reminisce about a fun-filled couple of days or simply to get a flavour of the *VB* conference in Toronto, see http://www.virusbtn.com/conference/. Copies of the VB2003 conference proceedings can be purchased from *Virus Bulletin* (either on CD or in printed format), along with conference backpacks and T-shirts while stocks last – email bernadette@virusbtn.com for details.

THE MENACE WITHIN

A recent report by the *Associated Press* claims that 'computer-savvy Romanians are fast emerging as a bold menace in the shadowy world of cybercrime', citing as part of its evidence the fact that more than 60 Romanians have been arrested in recent international operations concerning crime of an electronic nature.

While *VB* would advise strongly against making such blanket statements (and point out that the AV industry boasts many fine security experts hailing from Romania), *VB* was astonished that a spokesperson for Bucharest-based *BitDefender* made no attempt to refute the claim and, more surprising still, seemed to imply that the programmers within his own company would have been up to no good themselves but for having found gainful employment in the company. In reference to *BitDefender*'s programmers, Communications Manager Mihai Radu is quoted as saying, 'They can do anything. If they weren't working for us, who knows what they'd be up to.' Without wishing to stir up any trouble, *VB* wonders what an effect this statement to the press may have had on morale within the company – and fears it might take more than a round of chocolate biscuits to pacify a room full of affronted AV researchers …

Prevalence Table – September 2003

Virus	Type	Incidents	Reports
Win32/Sobig	File	52133	63.75%
Win32/Dumaru	File	12029	14.71%
Win32/Gibe	File	7320	8.95%
Win32/Opaserv	File	4399	5.38%
Win32/Mimail	File	1657	2.03%
Win32/Bugbear	File	873	1.07%
Win32/Dupator	File	767	0.94%
Win32/Klez	File	444	0.54%
Win32/Swen	File	372	0.45%
Win32/Yaha	File	239	0.29%
Win32/Nachi	File	216	0.26%
Win32/Funlove	File	181	0.22%
Win32/Lovsan	File	151	0.18%
Win32/Fizzer	File	135	0.17%
Win95/Spaces	File	124	0.15%
Redlof	Script	74	0.09%
Win32/SirCam	File	65	0.08%
Win32/Kriz	File	56	0.07%
Win32/Parite	File	53	0.06%
Win32/Magistr	File	35	0.04%
Win32/Ganda	File	32	0.04%
Win32/Valla	File	32	0.04%
Win32/BadTrans	File	28	0.03%
Win32/Hybris	File	26	0.03%
Fortnight	Script	23	0.03%
Win32/Lovgate	File	22	0.03%
Win32/Nimda	File	21	0.03%
Laroux	Macro	18	0.02%
Win32/Elkern	File	18	0.02%
Mumu	Script	17	0.02%
Win32/Deborm	File	17	0.02%
Win32/Spybot	File	15	0.02%
Others		181	0.21%
Total		81773	100%

The Prevalence Table includes a total of 181 reports across 63 further viruses. Readers are reminded that a complete listing is posted at http://www.virusbtn.com/Prevalence/.

VIRUS ANALYSIS 1

VIRUS MAPPING

Mikhail Pavlyushchik
Kaspersky Labs, Russia

The life of a virus analyst is tedious and monotonous. Worms, Trojan horses and viruses pass by in an endless queue. It's a very rare occurrence for a new malware class to appear and even rarer for a new platform to be infected. But in the last week of August 2003 the analyst's life became a little more interesting, with a new virus for a new platform. It was MPB/Kynel (initially announced as MBA/First).

But before the virus analysis I shall introduce you to *MapInfo*.

WHAT IS MAPINFO?

MapInfo is a 'Geo-Information system' – software used for mapping and geographic analysis. It is developed by *MapInfo Corporation* (http://www.mapinfo.com/). A *MapInfo* document consists of a database in which each record may contain user-defined data fields as well as graphic information (lines, circles, etc.). While saving a single document MapInfo creates several files:

.TAB	table structure in ASCII format (required)
.DAT	table data storage in binary format (required)
.MAP	storage of map objects in binary format (optional)
.ID	links to the .map file (optional, but required if .map file exists)
.IND	data of indexed fields in binary format (optional)

Alternatively, a document may be saved in MapInfo Interchange File (MIF) format, which is a single ASCII text file that fully describes the contents of a *MapInfo* table. As might be expected, files in this format have a '.MIF' extension.

MapInfo Professional has its own application development environment, MapBasic, which is a BASIC-like programming language. This is used to create custom applications for use with *MapInfo Professional* or special *MapInfo* runtimes. MapBasic extends geographic functionality, automates repetitive operations and integrates *MapInfo Professional* with other applications. With MapBasic it is possible to build a custom application by adding menu items to *MapInfo Professional* or even redesigning the entire user interface and extending the functionality of *MapInfo Professional*.

The MapBasic language is very similar to MS Visual Basic in syntax and functionality, but has a lot of extended statements for tables and map manipulations (over 300 statements and functions in total). For us, the most interesting functionality is as follows:

- Open, close, read, write to ASCII and binary files.
- Call routines from DLLs.
- Communicate with other applications using DDE.
- Control *MapInfo Professional* using DDE or OLE Automation.

The MapBasic compiler creates an executable file from a MapBasic program. The executable file has the extension '.MBX' (MapBasic eXecutable) and can be executed only with *MapInfo Professional* or special *MapInfo* runtimes. Compiled executables start with the following text header:

```
!App
!Version xxx
!Charset "CharsetName"
```

The first line is commonly used as a file format signature, but this line is case-insensitive and anti-virus scanners must be ready for this. Binary compiled code starts just after the header lines and has a fixed size. The rest of the file after the binary code is silently ignored by the run-time and thus may contain any data. MapBasic executables are cross-platform and may be executed on any platform supported by *MapInfo Professional* (*MS Windows*, Apple Macintosh, etc.).

VIRUS DETAILS

The virus code contains four procedures. The *Main()* procedure is the entry point for MapBasic applications. In this procedure the virus initializes some global variables. The interesting thing here is array initialized with string constants:

```
sMessage(1) = "M34)8(%!O<W138W)I<'0@1TE3(#$N,#`-" +
Chr$(34) + "C`S+S`U+S(P,#,@+2#DY>W\(/#N"
sMessage(2) = "KYN3E[>C_(.WNXN[I(/'E\.CH(.+H\//Q[N(-"
+ Chr$(34) + "L3@[>CK[N+S(._PZ.+E\@``"
```

This string constant is UUE-encoded text in Russian that may be translated as:

```
MBX PostScript GIS 1.00
03/05/2003 - new virus series birthday
Hi to Daniloff
```

The next two procedures are system event handlers. These procedures are called by *MapInfo* when appropriate events occur. The *WinChangedHandler* is called each time the user changes data in the map window or lists its content. The virus uses this handler to gain control multiple times while *MapInfo* is running. In this handler the virus performs two tasks – it infects the *MapInfo* environment and collects the filenames of tables.

The *EndHandler* is called on exiting the application. Here the virus runs its payloads and spreads itself.

The last virus routine, *EndMBX()*, is user-defined and contains only two lines of code:

```
Sub EndMBX()
    Set Handler EndHandler Off
    Set Handler WinChangedHandler Off
End Sub
```

It disables system event handlers WinChangedHandler and EndHandler. The virus calls this routine when some error has occurred to stop its further execution.

MAPINFO ENVIRONMENT INFECTION

The first thing the virus does on gaining control is to check for the existence of the '0gPiSs1.dll' file in the program folder (the folder in which the MapInfo executable is installed). If this file does not exist, the virus assumes it is running from an infected table in a clear environment, thus the .MIF file of the current table must contain virus code.

To make sure of this, the virus checks that the value of the first two bytes is equal to 16673 (0x4121= '!A') – which is the MapBasic Executable signature ('!App'). If this is not the case, the virus assumes '%applicationdir%\0gPiSs1.mbx' as source. It seems that the virus author used this file for the initial spread of the virus, because the virus itself never creates such a file.

Once the virus code has been found in the file (.MIF or .MBX) the virus copies the whole file into a file named '0gPiSs1.dll' in the program folder and inserts commands that run the virus code into the startup workspace file 'startup.wor':

```
!MBX PostScript 1.00
Run Application "s0gPiSs1.dll"
```

If the startup workspace file does not exist the virus creates a new one with the following text:

```
!Workspace
!Version 400
!Charset WindowsCyrillic
!MBX PostScript GIS 1.00
Run Application "s0gPiSs1.dll"
```

The startup workspace is loaded by MapInfo automatically on every start before any tables and even before any dialog box is displayed. While interpreting the workspace file MapInfo executes the 's0gPiSs1.dll' file, regardless of its extension – thus the virus gains control before any tables have been loaded.

SPREADING

From the virus writer's point of view there is no point in modifying tables files at the time WinChangedHandler is called because they will be overwritten by *MapInfo* on closing. So the virus cannot infect tables here, but it can collect tables' filenames and it performs propagation later – on *MapInfo* closing.

Each time WinChangedHandler is executed, the virus enumerates opened tables and collects those which it has not infected yet. As an infection mark the virus uses the '\GIS' key in the table metadata, with value 'PostScript'.

On exiting the application the EndHandler procedure gains control. Here, for each collected (i.e. not infected) table the virus overwrites the .MIF file with its own code. The virus does not use MIF format at all, it just uses the extension to masquerade as a standard *MapInfo* file. In cases where the file size is greater than the virus code the rest of the file is left unchanged (containing text data).

Now the code is ready, and the virus has to infect tables (.TAB files). First the virus ensures all tables are closed (with the *Close All Interactive* statement) to avoid file overwriting, then it inserts the following lines into each table:

```
!MBX PostScript 1.00
Run Application "%pathname%\%tablename%.mif"
```

While opening such a table *MapInfo* silently executes the application from the .MIF file and the virus gains control.

PAYLOAD

The payload of the virus has two similar parts and activates depending on the system date. On Monday the virus runs the first part of its payload, which with a probability of 1% for each collected (in WinChangedHandler) table, tries to delete files with extension .map, .tif, .pcx or .jpg. The second part of the payload triggers on Friday 13 and does the same as the first part, but with a 14% probability. In addition it overwrites the 'mapinfow.prj' file with text written in Russian, which may be translated as:

```
"– Coordinates –"
"Longitude / Latitude", 3, 62, 8, -74, 40.5,
40.6666666667, 41.0333333333, 2000000, 100000
```

CONCLUSION

Of course this threat has no chance of climbing to the top of any virus prevalence tables, but its successors may become a huge headache for *MapInfo* users. The problem here – *MapInfo* users never think about viruses while exchanging maps and tables or downloading utilities in the form of MapBasic executables from the Internet. It was safe before, but not now… The best way I see to prevent future threats is to implement security features based on digital signatures in *MapInfo* products, as is done in *Microsoft Office* – and of course the security features must be ON by default.

TECHNICAL FEATURE

VIRUS CRYPTOANALYSIS

Mircea Ciubotariu
BitDefender, Romania

'A soul on his journey to the Afterlife comes to a fork in his path. One way leads to Heaven and is guarded by an angel who always tells the truth; the other way leads to Hell and is guarded by an angel who always lies. The soul is permitted to ask only one question before he must continue his journey. 'If I were to ask your buddy here which is the way to Heaven what would he reply?' he asked one of them. Given the answer he knew for sure which path to follow …'

Riddle from a recreational math book

INTRODUCTION

As technology has evolved, more opportunities have become available for virus writers to express their imagination in malicious code.

The introduction of cryptography into virus writing has become a necessity in order for virus writers to protect their code against external factors that might reveal its malicious intentions – such as anti-virus programs.

In general, matters concerning cryptography reside in the time required to encrypt/decrypt the information compared to the strength of the algorithm used. Any functional encrypted virus has to decrypt its code in order run it, whether it decrypts instruction by instruction, as DarkParanoid did (see *VB*, January 1998, p.8), or even if it attempts brute force on its encrypted part, like DarkMillennium (W32/Darkmil).

Encryption is performed by applying a function to the original viral code and transforming it into a chunk of data which becomes meaningless without the decryption function and encryption key(s).

Decryption of the encrypted chunk of data is achieved by applying the inverse of the function used to encrypt it, but with the same key(s).

Keys are the initial parameters used by encryption functions. Generally the keys are a byte or a word in size for 16-bit viruses, while the keys for 32-bit viruses are a double word.

A clear distinction must be made between the decryption code and the encrypted virus body. Decryption code in its essence is not harmful and acts as a tool attached to the virus, but in many cases it may be considered to be a signature of a specific virus due to the information contained within, such as the encrypted block length, relative addresses, etc.

That is why in some cases it would be relatively safe to extract a signature for a virus from its decryption code, but not foolproof.

Many existing encrypted viruses use very simple encryption functions and hence can be caught using a CPU emulator which goes beyond the encrypted layer(s) within a relatively short amount of time by emulating a given, specific, number of instructions.

Problems arise when the length of time spent emulating in order to catch an encrypted virus is too long, or when by natural means the virus damages its decryption code – or simply when the information contained in the decryption code is useless.

THE CONCEPT

Because many encrypted viruses use simple encryption functions such as ADD or XOR, we shall consider a different approach to virus recognition and try to adapt the notion of the virus signature in the light of the concept explained below.

Let us suppose we are dealing with a file infector virus which uses a simple function to encrypt itself. Each time it infects a file it encrypts its main body in a buffer with a random key, then attaches the decryption code, set up with the encryption key, to the victim file and finally it appends the encrypted data.

Now let us drop all the information contained in the decryption code and remain only with the encrypted data. This means we know nothing about the function or the key that was used to encrypt it.

Next, we select one of the more commonly used functions (let's say ^ and its inverse ~). Even if the function has been guessed correctly we still need a key in order to get any valid information from our chunk of data.

Labelling the units (bytes, words or double words, according to key size) in data chunks as A_0, A_1, A_2, … in order (e.g. A_0 is the first unit, A_1 the second etc.), we assume that they were obtained by performing the function ^ on the original units a_0, a_1, a_2, … with the key K.

For the sake of simplicity we assume for this example that the key was kept constant during encryption.

We have:

$A_0 = a_0 \wedge K, \quad a_0 = A_0 \sim K$
$A_1 = a_1 \wedge K, \quad a_1 = A_1 \sim K$
$A_2 = a_2 \wedge K, \quad a_2 = A_2 \sim K$
…

Assuming the function has the associative property let us consider:

$A_0 \sim A_1 = (a_0 \wedge K) \sim (a_1 \wedge K) = a_0 \sim a_1 = s_1$
$A_1 \sim A_2 = (a_1 \wedge K) \sim (a_2 \wedge K) = a_1 \sim a_2 = s_2$
…

For easier understanding, imagine ^ is XOR function, so ~ is the same as ^.

Thus, for any *n* given units of data where $n < N$ (*N* is the total number of units in chunk), we have $(n - 1)$ units of s. This is a transformation of our initial block of data independent of encryption key at a cost of one unit.

The resulting $(n - 1)$ units block may be considered as a hash value which can be looked up in a table of such hashes. If the hash is found, further comparison is performed based on the same criteria.

If the function we chose was wrong (i.e. the hash value was not found in the function hashes list), we try other functions until we have either a match or there are no more functions remaining to test.

POLYNOMIAL FUNCTIONS

This approach considers and is limited to ADD and SUB functions taking into consideration a key modifier K_1 for K_0; that is at each step of encryption another function ADD or SUB is applied to the key K_0 with parameter K_1.

Let us refer to the encryption function as + (SUB is the same as ADD with negative argument) and we have:

$A_0 = a_0 + K_0, \quad K_0' = K_0 + K_1$
$A_1 = a_1 + K_0', \quad K_0'' = K_0' + K_1 = K_0 + 2K_1$
$A_2 = a_2 + K_0'', \quad K_0''' = K_0'' + K_1 = K_0 + 3K_1$
…

The equivalent polynomial function is $f(x) = K_0 \cdot x^0 + K_1 \cdot x^1$, where x is the current step in encryption (or unit index). We shall consider:

$A_0 - A_1 = (a_0 + K_0) - (a_1 + (K_0 + K_1)) = a_0 - a_1 - K_1 = s_1$
$A_1 - A_2 = (a_1 + (K_0 + K_1)) - (a_2 + (K_0 + 2K_1) = a_1 - a_2 - K_1 = s_2$
…
$s_1 - s_2 = (a_0 - a_1 - K_1) - (a_1 - a_2 - K_1) = a_0 - 2a_1 + a_2 = S_1$
$s_2 - s_3 = (a_1 - a_2 - K_1) - (a_2 - a_3 - K_1) = a_1 - 2a_2 + a_3 = S_2$
…

We get $(n - 2)$ units of S, which are an exact transformation of the original data block, independent of the key and key modifier.

It is possible to have another modifier, K_2, for the K_1 modifier, but in practice this situation is very rare and the solution would be to iterate the above once again. The general form of polynomial function of the *n*th degree is:

$f(x) = K_0 \cdot x^0 + K_1 \cdot x^1 + \ldots + K_{n-1} \cdot x^{n-1} + K_n \cdot x^n$

IMPLEMENTATION

At implementation level this may be accomplished in two ways: we either use key-independent hashes in signatures or keep a long enough hash from the original signature bytes (decrypted) and with it generate key-independent hashes at run time.

The first solution is limited in that it applies strictly to the data encrypted with the same function that was used to generate the signature – so viruses that use a random function from a given set require as many signatures as encryption functions used. Another limitation of the first solution would be that, even if the signature is identified correctly, the decrypted data still won't be available, since the trick of this type of hash is to avoid keys.

Therefore the second solution may be a more efficient implementation, especially because it gives the whole decrypted data and the key used, as:

$K = A_0 \sim a_0 = A_1 \sim a_1 = A_2 \sim a_2 = \ldots$
$x = X \sim K$

Where K is the key deduced from the hash, X is any unit of data outside the hash. Using K the whole chunk of data is decrypted and further classic methods may be applied for an exact match or for further analysis.

CONCLUSION

The cost of achieving cryptoanalysis as described here is a slight decrease in performance due to the run time generation of key-independent hashes and also a slight increase in the amount of storage space required for signature(s) and for each function implemented as many look-up tables are required.

However, it should be mentioned that this approach comes as an extension of the classic hash signature type which may be regarded as encrypted with a function f(x) = x, where x is the original data.

Finally, the ideas presented in this paper are just a starting point for what may become the basis for a powerful and more complex cryptoanalysis engine as other functions such as ROR or ROL may be easily implemented. Although mixed functions and multi-layer, multi-function encryption methods push the complexity beyond practical implementation, these are subject to a different approach.

CONFERENCE REPORT

TOTALLY TORONTO: VB2003

Helen Martin

As the plane touched down in Toronto, the *Virus Bulletin* crew felt a certain sense of déjà vu: through the plane's windows, all that could be seen was torrential rain and a flooded runway. On countless occasions over the course of the last year we felt we could have kicked ourselves for having tempted fate as blatantly as with the closing slide of VB2002: '*See you in Toronto – come if you dare …*'

Toronto seemed like a perfectly 'safe' location for a conference but, as the organisation of the conference progressed, the world's media reported the outbreak of SARS in Toronto – and when the W.H.O. issued an advisory against travel to the city, the future of VB2003 hung in the balance. However, *VB* remained confident that Toronto would recover quickly and our optimism was rewarded when the city was declared a safe destination only a couple of weeks later.

A massive power cut across a large part of North America was the next to wobble our nerves and, in the week leading up to the conference it was with disbelief that we heard reports of a hurricane travelling toward the East coast of North America, and heading inland. But, despite initial concerns, the rain on arrival in Toronto was about as far as the similarities to VB2002 would go and VB2003 proved to be possibly one of the smoothest-running *VB* conferences on record.

THE FULL FAIRMOUNTIE

This was *VB*'s second visit to Canada, and the welcome was every bit as warm as the first. The grandeur of the Fairmont Royal York provided the perfect setting for the 13th *Virus Bulletin* conference. Characters of legend looked down on dining delegates from the magnificent hand-painted ceiling of the Ballroom, while the two conference halls were spectacularly ornate (and a healthy distance between the halls ensured that delegates had truly earned their chocolate cookies by coffee break).

Wot, no disasters? The relieved VB team.

An exhibition was set up in the spacious hallway between the two conference rooms and featured booths from *CA*, *Eset*, *Sophos*, *NAI*, *ICSA Labs* and *Virus Bulletin*. A caricaturist made an entertaining addition to the *NAI* booth and was kept busy by a constant stream of subjects waylaid en route between conference sessions.

As usual a drinks reception was held on the eve of the conference. Conversation, beer and wine flowed freely and our bona fide Mountie was kept busy talking to and posing for photographs with *VB* delegates. In fact, such was the draw of RCMP Allen Rodgers in his eye-catching uniform, that a stream of delegates from a different event sneaked along the corridor to have their photographs taken with him too.

Now, that's magic!

After the Canadian-themed welcome drinks, the entertainment for this year's gala dinner was on a magical theme. British magician David Penn made his way around the dinner tables amazing delegates with his award-winning close-up magic. His stage act followed and, for half an hour, all eyes in the room were glued to the stage while he performed the seemingly impossible. After dinner there was plenty to talk about as the naturally analytical minds of AV experts battled with the frustration of not being able to answer to the question: 'how *did* he do that?'. Theories were in abundance, but David Penn wasn't giving anything away.

THE PROGRAMME

With a wide range of AV-related subjects on the programme, delegates had plenty to choose from – indeed, some were heard lamenting the fact that they were not able to be present in both streams at the same time.

In the corporate stream, presentations by Chuck Springer and Jeannette Jarvis provided real-world examples of how large corporations deal with threat assessment and incident management. David Phillips outlined the reasons for setting up a new Open University course on 'vandalism in cyberspace', while David Perry proclaimed that it is not user education that is needed, but user understanding.

Bruce Hughes brought delegates up to date on *ICSA Labs*' progress with the Real-Time WildList. The project has involved developing a system which will allow WildList reporters to submit virus reports weekly, daily or even hourly. Bruce anticipated that HTTPS upload for sample submission should be available by the end of 3Q 2003 and that an online database will be ready by the end of 4Q 2003.

In the technical stream, Frédéric Perriot discussed his research into the use of code optimization techniques in

Richard Ford, James Wolfe and Shawn Campbell discuss their frustration with virus naming.

dealing with polymorphic viruses. Kurt Natvig showcased the use of the sandbox described in his papers at VB2001 and VB2002 to demonstrate the capabilities of real-life virus samples. Martin Overton picked up from where John Morris left off at VB2002 and described his own use and development of John's SMB-Lure design, while Neal Hindocha and Eric Chien impressed the audience with a practical demonstration of vulnerabilities in instant messaging.

VB2003 saw the introduction of a new style of panel discussion. The single vs multiple engine debate involved six AV vendor representatives debating the merits of single engine and multiple engine scanning methods. Thankfully disagreements were confined to the 40-minute session and intervention was not required to end any squabbling.

At the close of day one, a discussion panel was held on the subject of anti-virus testing. Panel members Michael Parsons (*West Coast Labs*), Larry Bridwell (*ICSA Labs*) and Matt Ham (*VB*) explained their current AV testing practices, outlining the factors that limit the ways in which they can test products. The panel members concurred that, given unlimited resources and time they would carry out a range of tests that would challenge aspects of anti-virus products *other* than their in the wild detection rates.

The traditional close of conference panel discussion was devoted to the emotive subject of virus-naming. It wasn't quite the virus-naming discussion to end all virus-naming discussions, but it provided the opportunity to hear from AV researchers (including CARO members) and from real-world end users who experience regular frustration and difficulties as a result of the lack of virus-naming standards.

AND SO TO VB2004 ...

VB2003 was blessed with a very enthusiastic and faultlessly professional team of organisers, helpers and audio-visual crew, without whom the event would not have run as smoothly. Thanks are due to all those who helped put the conference together, to the conference sponsors and, of course, to all of the VB2003 speakers and panellists.

Not ones to rest on our laurels, we hope to build on the successes of VB2003 to make VB2004 a better event still. And now to the burning question: where will VB2004 be held – Zanzibar, Dublin, Baghdad, Chicago, Casablanca or Kathmandu? While contractual issues currently keep the location of VB2004 under wraps, all will be revealed shortly. Watch http://www.virusbtn.com/ for details.

FEATURE

WHERE TO FROM HERE?

Myles Jordan
Computer Associates, Australia

Not long ago, I was asked by a journalist what I thought was the future of malware and the anti-virus industry. This struck me as a simple question, with a remarkably complex answer.

The difficulty in predicting the future arises from the fact that what will occur in the future is decided by the cumulative actions of a number of groups and individuals, many of whom may be working in opposition to one another. This is especially true of many arenas – war and politics, to name just two. The strange dance of those working for and against the prevalence of malware can also be included in this bracket.

It is a strange dance indeed, the to-ing and fro-ing of the malware creators, as they find yet another technological or sociological vulnerability to exploit, and the security researchers, as they create and distribute tools to secure the latest breach. But will this dance continue? Fundamental changes that affect malware have been occurring for some time now: will they result in some permanent shift in the status quo?

REVELATIONS IN MALWARELAND

One of these fundamental changes has been the deceptively obvious revelation that there is an abundance of ways in which malware can be propagated.

The classic viral strategy is to attempt to infect as many files on a local machine as possible, hoping that one or more of those files is copied to another machine, and accessed there. History has shown this to be very inefficient, relative to the method of using the local machine as a launch pad from which attempts can be made to infect other machines directly, via a network connection.

This popular method typifies a sub-category of viruses: worms. In contrast to viruses, which attempt to infect all relevant files on a computer, worms attempt to infest all relevant computers on a network. The method of infestation used by a worm is irrelevant; all that matters to its propagation is that the machine is infested, and can thus be used to help infest other machines. While file viruses exploit internal vulnerabilities within the computer, such as limited restrictions on file modifications, worms tend to work by exploiting vulnerabilities in defensive barriers external to the computer.

The significance of this realisation is that the current trend of the creation of more worms and fewer file viruses is

likely to continue into the foreseeable future. This trend seems to be running in parallel with another, which is a change in the type of programming languages used to create malware.

In the early days of malware, when virtually all were file viruses, the programming language most commonly used to create malware was assembler. This continued for as long as file viruses maintained their popularity, probably because the attention to detail that is fundamental to assembler programming is implicit in the task of intertwining viral code with clean code during the process of infecting a file.

The task presented to a worm, on the other hand, tends to be more abstract and less detailed; infesting a system by copying one or more files and perhaps changing some registry keys does not compare to the intricacy of fiddling with file headers and code redirectors.

OF BARRIERS AND VULNERABILITIES

Throughout human history, we have used barriers to protect our assets. These have taken many different forms, including physical barriers such as fences, and psychological barriers such as intimidation.

Today, to protect our computer-based electronic assets we use physical, technological and sociological barriers. While we use the physical barriers to defend against traditional threats such as theft and vandalism, we use the technological barriers (such as anti-virus software and firewalls) and sociological barriers (such as common sense and scepticism) to defend both the integrity and intent of our computers. It is with these barriers and their vulnerabilities that we are concerned.

Common sense tells us that the best way to defeat a barrier is to bypass or avoid it completely – for example walking around a fence. The most common vulnerabilities in a defence system are not necessarily in the barriers themselves, but are much more likely to exist as a result of gaps between the barriers.

TECHNOLOGICAL BARRIERS AND VULNERABILITIES

In the digital world, computer hardware can be thought of as the tool, and software can be thought of as the intent, or purpose, for which that hardware is used. Similarly, where a wall could be considered a tool, a barrier can be thought of as the intent.

Accordingly, virtually all technological barriers are some form of software. Separately, software barriers perform many different functions and defend against many different attack vectors; they are the front line of defence. It is against these barriers that the majority of attacks are launched.

Fortunately, a fairly complete range of technological barriers exists, including firewalls and intrusion detection systems intended to defend against network penetrations, anti-virus gateways used to prevent malware from reaching a machine, and local anti-virus software intended to negate any malware that may circumvent the external barriers and succeed in accessing a local machine.

That such a complete range of defences is available is a Good Thing™, but there are still vulnerabilities in the system that can be exploited during a malware attack.

The most dangerous of these vulnerabilities is, of course, the absence of any of the fundamental defensive pieces – for example a machine whose anti-virus signature files are obsolete, or which does not have a firewall. While these are not common situations among business computer networks, many home users' machines run without firewall software installed, which can lead to home machines being used as launch pads for sustained malware propagation attempts.

The other major type of vulnerability is the software vulnerability. This term groups together deficiencies in software such as buffer overflow bugs, particular design flaws, and so on.

The problem with these software deficiencies is that they can constitute a vulnerability in the defences of a computer system. This sort of vulnerability usually means that an attacker who has gained some level of access to a machine could gain unintended privileges on, or further access to, the machine. In the worst case, a severe vulnerability could be exploited by an attacker, allowing the execution of arbitrary code, thus subverting the intent of the system.

WHY DO TECHNOLOGICAL VULNERABILITIES EXIST?

Surely, given the level of damage made possible by technological vulnerabilities alone, no software vendor would release software that contained any vulnerabilities, would they? Unfortunately, it is not that simple, and many factors are involved with the creation and continued existence of technological vulnerabilities.

To begin, there is the already vast, yet still increasing, complexity of software itself. This seemingly implacable trend is driven by a number of factors, including users' continual desire for new features, and software vendors who release new versions in order to generate new sales, to name just two.

Indirectly, the increasing complexity of software also leads to the creation of ever higher-level languages meant to

facilitate the inclusion of more new features, with less effort on the part of developers.

While these new languages may temporarily simplify tasks for developers, all that is really happening is that the increased complexity is concealed by the language's compiler, from whence it will be multiplied by the re-increasing complexity of the developer's code.

The problem with complexity, of course, is that the more there is of it, the greater the chance that some deficiencies will slip past the testing routinely performed by vendors and into a released product, and then the deficiencies could become real vulnerabilities.

Many major software packages have lists of past and present bugs and other deficiencies numbering in the hundreds of thousands, or more. And those are just the known ones.

To be fair, many of the past bugs have been fixed, and a great majority of current bugs are minor and will never constitute any form of security risk. Still, this helps illustrate that increasing complexity can lead to exponentially increasing numbers of software flaws. The various organisations that monitor active vulnerabilities maintain lists with thousands of entries.

When a software vendor realises that their software contains some sort of vulnerability, they will usually create and release a patch or update that users can apply to neutralise the vulnerability. Unfortunately, a number of significant hurdles exist between the introduction of a vulnerability and the eventual application of a patch.

First, the deficiency has to be discovered and reported, which can take days, weeks, months, or even years. If the vulnerability happens to be discovered by someone with malicious intent, they can exploit it freely for as long as it goes unreported.

Once the vulnerability has been reported, it must be analysed by the manufacturer of the software in order to discover the flaw that causes the vulnerability, and to work out what is required to fix it – again a task which may take days or weeks to complete.

Once the manufacturer has produced a patch to correct the flaw and made it available to their users, it might seem that the issue is closed. Unfortunately, however, there is a further issue to be dealt with, which is who actually applies the patch.

Vendors cannot force users to apply patches, even critical ones. That would be invasive, possibly even illegal. All a vendor can do is advertise in whatever way they see fit the fact that the patch is available, and then it is the user's responsibility to obtain and apply it.

Unfortunately, it is often the case that a majority of users will not be actively informed of the existence of the patch, let alone of the vulnerability, and thus are forced to check for any new patches themselves. But even if they were all informed of the existence of the patch, the problem does not end there: many people would ignore the notification through ignorance, arrogance or laziness. Some may be too busy and delay the application of the patch until they forget. Others may feel they could not afford the down-time required to apply the patch, and others yet may mistrust the patch itself, believing it may introduce new flaws.

The emergence of semi-automatic patching systems (i.e. ones that do all the hard work for the user, but request the user's permission before updating anything) tends to help the situation, but these walk a fine line between supplying relevant updates and nagging users, who might then disable them. Nor do they address the remaining issues of lost productivity, downtime, or possible new flaws.

SOCIOLOGICAL VULNERABILITIES

Even if all computers the world over had the complete gamut of defensive software, and that software was fully patched and free from vulnerabilities, our computer systems would not be immune to externally inflicted harm. This is because technological vulnerabilities are not the whole story.

No matter how impenetrable a computer's defences are to purely technological assault, the fact that computers are controlled by imperfect humans means that sociological vulnerabilities will exist and be open to exploitation.

Humans are in positions of great power in relation to the computer systems they manage, and this means that any sociological vulnerability that can be exploited during an attack will commonly allow the attacker to bypass any existing technological defences, because, in effect, the attacker has gained a level of power equivalent to that of the exploited user.

Given that so much power is available, it should come as no surprise that attempts to exploit human weaknesses as security vulnerabilities are very common.

The majority of successful worms have used email messages as a form of propagation, and the most successful of these have tended to be those that have used superior social engineering techniques to convince the human on the receiving end of a worm-laden email to activate it.

Email systems are the perfect stage for those wishing to exploit sociological – as opposed to technological – vulnerabilities. Email systems have little or no inherent technological defences, because they are modelled on how

humans prefer to communicate: without obstruction or hindrance, and all security systems involve some sort of obstruction.

The only sociological defence is human moderation, which is too costly to implement broadly. Without any implicit defences, email provides direct access to what is commonly the weakest link in the security chain: the human user.

The exploitation of human emotional needs has been going on for millennia, and it is likely to continue for as long as humans are around. The problem, of course, is that we are fundamentally emotional creatures. Once it is realised that a person's emotions can be exploited to manipulate their actions, a powerful tool is created that can be used by the unscrupulous amongst us.

A well known example of this sort of emotional manipulation is the so-called 'Love Bug' worm (aka VBS/LoveLetter@mm). This was a relatively simple worm, in that it contained no advanced anti-technological barrier measures, yet by merely pretending to be a love letter sent to the unsuspecting recipient, it managed to become one of the most widespread pieces of malware of all time.

The success of Love Bug stems from its overt exploitation of the all-too-human desire to be loved. By at least appearing as though it was about to deliver a love letter, it managed to fool the recipient into dropping their guard, thus allowing the worm to perform its function unhindered. Its function, of course, was to send itself, via the recipient's list of email addresses, on to new victims who would, in turn, drop their guard, allowing it to spread further, and so on.

The technique used by Love Bug and many other worms to exploit sociological vulnerabilities is part of a group of such techniques that are collectively labelled 'social engineering'.

Social engineering techniques attempt to exploit common human weaknesses such as greed, the desire for love, situational ignorance, or the desire for success. Such techniques achieve their goal by promising to satisfy one or more of these fundamental desires in exchange for some action on the part of the victim.

After the success of such worms, users are very slowly learning to be more cautious about deciding which email attachments to open; if something appears unsolicited, and looks unusual, many users will discard it, or at least double-check with the supposed sender as to the validity of the message.

This increased awareness of such security threats, and the resulting alertness amongst users to unusual emails, has caused a noticeable, but not yet predominant, shift towards more subtle social engineering techniques, in particular those that attempt to slip under the target's unusual situation detector.

Unlike the more obvious social engineering techniques that blatantly attempt to entice their target, worms using this type of approach actually attempt to appear wholly innocuous. The idea is that the target will not become suspicious, and therefore may just thoughtlessly do what is asked of them.

These techniques rely upon the fact that humans tend to assign a level of trust to people with whom they are acquainted, and so will be more likely to believe something to be innocuous if it came from an acquaintance rather than from a stranger. Once this artificially heightened level of trust has been gained, it is much easier to manipulate the target because they believe the instructions came from a trusted source.

Given that virtually all email worms attempt to propagate to email addresses obtained from the current host's list of contacts, most email worms have already successfully utilised the first part of this social engineering technique: appearing to arrive from a trusted source. All that is subsequently required for the success of the worm is to appear sufficiently innocuous that the recipient does not become suspicious – yet at the same time sufficiently compelling to make the recipient open it.

The technique of being simultaneously innocuous and compelling was demonstrated recently by the Sobig.F worm. The technique was utilised so well by Sobig.F that, at the time of writing, it has been the most widespread email worm ever.

CONCLUSION

Many people are beginning to use computers for the first time, and these people have an understandably poor grasp of what may represent a threat to them.

Security companies, individuals in the AV industry, and relevant media outlets, continually make attempts to educate users regarding the security risks that seem to have become a part of modern computing.

The knowledge they disseminate is intended to provide users with a basic understanding of the threats posed by malware and the realities of software vulnerabilities and how to deal with them. If users could gain the ability to recognise, and thus defeat, social engineering tricks, it would be of enormous benefit, and not just in relation to computer security. It is hoped that future generations of computing novices will not need to be taught what are safe practices and what are not – they will just know.

COMPARATIVE REVIEW

WINDOWS 2003 SERVER

Matt Ham

Another comparative review and another new operating system enters the *VB* labs to be overloaded and treated far worse than perhaps it deserves. New OSs are always good for revealing bizarre new faults in previously unbreakable pieces of anti-virus software. The areas in which novelty has previously caused difficulties have been with the on-access scanners, and in particular, floppy disk scanning on access. Having experienced red faces in the past, I imagined developers would have become somewhat more careful on these matters, though some slip-ups were expected to unfold as the review progressed.

Windows 2003 Server itself was relatively pleasant to work with. One feature I found intriguing was that, under *Windows 2003 Server*, anti-virus products are expected to be able to run simultaneously. Various developers have mentioned this feature as having worked to a greater or lesser extent in their ad hoc tests – which is a mixed blessing. If 95 per cent of two product combinations work simultaneously (which is what most results seem to indicate), this may encourage people to run two on-access scanners on mission-critical machines. This may delight the 95 per cent, but there will be gnashing of teeth for those who discover themselves amongst the incompatible 5 per cent. This feature did inspire some crazy thoughts of using just one test machine to perform all the comparative tests simultaneously – though this is perhaps something I shall suggest other reviewers do, while sniggering quietly to myself.

Although the deadline for product submissions for this test was 6 October, the test sets were based upon the July 2003 WildList – a long delay indeed. However, this was the newest data available at the time (blame for which may partially be laid at the door of the *Virus Bulletin* conference for having dragged the WildList team away from their desks). Although new WildList data was available three days after the deadline, this data was not used as it was from the newly inaugurated Real-Time WildList. In future reviews *VB* plans to use WildList data from approximately 24 hours before the comparative deadline. There are still some issues to be decided, since this will make geography a real factor in the submission of products – but expect more potential for failure in the months ahead.

As for additions to the test sets this month, there were rather more than the usual bunch and a selection across the various types. Most unusually, a batch virus, BAT/Mumu.A, made its way into the wild, as well as the slightly more common sprinkling of macro viruses. Of particular note when replicating these new samples were the number now attempting to use peer to peer networks as a form of propagation. This feature leads to huge numbers of supposedly tempting-sounding files lurking in folders. While I admit that some individuals might be tempted by Jennifer Lopez engaged in amorous pursuits with a lavatory, one would hope that a user seeking complete guides to PHP4 might realise that these are unlikely to be distributed as .EXE files via *Kazaa*. Such diversions aside, the test sets looked to contain few horrors for the products and a bumper crop of VB 100% awards was anticipated.

AhnLab V3Net SE SP2

ItW Overall	100.00%	**Macro**	98.18%
ItW Overall (o/a)	100.00%	**Standard**	85.42%
ItW File	100.00%	**Polymorphic**	45.48%

While *V3Net*'s pastel colour scheme looked somewhat out of place on the staid desktop of *Windows 2003*, its performance did not suffer. Misses were slightly higher on access than on demand, but there were no misses in the wild. However, the treatment of infected items was rather bizarre. In most cases objects are deleted or disinfected through the use of a dialog box which pops up. I was unable to predict quite when this box would pop up and there were similar arbitrary delays in infection reports reaching the log file. As a general rule, the older the virus, the less chance there is of *V3Net* detecting it. Thus, while detection of older macro, standard and polymorphic viruses was relatively poor, newer threats were well detected. This selective detection pays dividends elsewhere, with very fast scanning rates and a zero false positive rate – which adds up to a VB 100% award for *AhnLab*.

Alwil Avast! 4.1.29

ItW Overall	99.58%	**Macro**	99.56%
ItW Overall (o/a)	N/A	**Standard**	99.73%
ItW File	99.56%	**Polymorphic**	91.21%

Avast! was the most obvious victim of the move to a new OS, with mysterious service failures appearing after installation. The program also declared that the virus vault and resident protection job were being supplied with null initialisation data and thus failed to operate. This made the on-access portion of the program impossible to test. In addition to these problems there was another reason for the product missing out on a VB 100%, with BAT/Mumu.A being missed in the ItW set. However, detection was otherwise good and the product continues to be enhanced with each release. It was a moment of joy indeed when I

On-access tests	ItW File		ItW Boot		ItW Overall		Macro		Polymorphic		Standard	
	Number missed	%	Number missed	%	Number missed	%	Number missed	%	Number missed	%	Number missed	%
AhnLab V3Net	0	100.00%	0	100.00%	0	100.00%	81	98.11%	9239	42.74%	314	85.38%
Alwil Avast!	-	-	-	-	-	-	-	-	-	-	-	-
CA eTrust Antivirus	0	100.00%	0	100.00%	0	100.00%	4	99.90%	1	99.89%	2	99.88%
CA Vet Anti-Virus	0	100.00%	0	100.00%	0	100.00%	0	100.00%	2	99.87%	4	99.78%
CAT QuickHeal	0	100.00%	0	100.00%	0	100.00%	107	97.45%	1086	92.85%	660	60.88%
DialogueScience Dr.Web	0	100.00%	0	100.00%	0	100.00%	0	100.00%	0	100.00%	0	100.00%
Eset NOD32	0	100.00%	0	100.00%	0	100.00%	0	100.00%	0	100.00%	0	100.00%
FRISK F-Prot Antivirus	0	100.00%	0	100.00%	0	100.00%	0	100.00%	3	97.53%	3	99.79%
F-Secure Anti-Virus	0	100.00%	0	100.00%	0	100.00%	0	100.00%	1	99.92%	3	99.85%
GDATA AntiVirusKit	0	100.00%	0	100.00%	0	100.00%	0	100.00%	0	100.00%	0	100.00%
Grisoft AVG	0	100.00%	0	100.00%	0	100.00%	23	99.44%	925	81.40%	47	97.15%
Kaspersky KAV	0	100.00%	0	100.00%	0	100.00%	0	100.00%	1	99.92%	11	99.69%
MicroWorld eScan	0	100.00%	0	100.00%	0	100.00%	0	100.00%	1	99.92%	11	99.69%
NAI McAfee VirusScan	0	100.00%	0	100.00%	0	100.00%	0	100.00%	0	100.00%	3	99.79%
Norman Virus Control	3	99.67%	0	100.00%	0	99.68%	-	-	-	-	-	-
NWI VirusChaser	1	99.56%	0	100.00%	1	99.58%	4	99.90%	0	100.00%	4	99.69%
SOFTWIN BitDefender	0	100.00%	0	100.00%	0	100.00%	13	99.69%	22	96.55%	40	98.88%
Sophos Anti-Virus	0	100.00%	0	100.00%	0	100.00%	8	99.80%	18	98.06%	14	99.49%
Symantec SAV	0	100.00%	0	100.00%	0	100.00%	0	100.00%	0	100.00%	0	100.00%
Trend ServerProtect	0	100.00%	0	100.00%	0	100.00%	0	100.00%	215	95.77%	1	99.98%
VirusBuster VirusBuster	0	100.00%	0	100.00%	0	100.00%	0	100.00%	101	91.45%	11	99.67%

realised that the extremely long registration keys have been replaced by a registration file. Sadly, however, constant improvements are likely to be the cause of the new problems.

CA eTrust Antivirus 7.0.139

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

Having stumbled somewhat in the last comparative review, *eTrust Antivirus* returned to its customary excellent detection rates this month. The patching process has always been a part of *eTrust*'s installation, and in this case three patches and two updates were required for installation. However, the patching process was rather more automated than previously, which made this task much less arduous. Unfortunately, not all change is good, and this was most obvious in the logging features of *eTrust*. These are now entirely binary while stored, with the exported versions being formatted in such a way as to render them all but useless. Only the very small number of missed samples made it feasible to use the logs at all – by selecting the toggles for displaying misses only, and reading the results from the screen. More disturbing than this, however, was the

sudden arrival of false positives in the clean set, which denied *eTrust* a VB 100%.

CA Vet Anti-Virus 10.59.2

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

Matters looked hopeful for *Vet*, as it was the only product of those reviewed to declare itself loudly as being a specifically *Windows 2003* product. *Vet*'s detection rate for polymorphics was vastly improved over past results. This does not seem to have added any huge slowdown for the clean set timings and neither were any false positives noted. The improved detection rate included a full detection in the ItW test set, thus earning *Vet* a VB 100% award.

CAT QuickHeal X Gen 7.0

ItW Overall	100.00%	**Macro**	97.54%
ItW Overall (o/a)	100.00%	**Standard**	82.60%
ItW File	100.00%	**Polymorphic**	92.89%

QuickHeal is another product opting for the non-detection of a variety of older viruses. Of particular note is the much higher detection rate in the standard test set when on-demand rather than on-access scanning is performed. The same trend is seen in the standard and macro virus sets, though not to the same extent. This does make some logical sense in that the undetected viruses in question are very unlikely to be able to operate on a modern machine. Full detection of ItW viruses, coupled with a distinct lack of false positives, means that a VB 100% is awarded to *QuickHeal*.

DialogueScience Dr.Web 4.30

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

Dr.Web achieved full detection over all the test sets and consequently earns a VB 100% award. Of note was a new feature that had crept into the *Dr.Web* installation routine. What I took at first to be a very short licence affirmation was in fact a declaration that no other anti-virus program was running on the machine. Without this the installation would not proceed. Though this should be an unnecessary precaution on *Windows 2003* it is certainly a useful method of encouraging users to be more careful when installing products. Also of note was the flagging of suspicious files. Only one file was flagged as suspicious, but when scanning the same files in zipped form the number of suspicious files increased dramatically. This is not surprising given the predilection of viruses to use multiple encryption and packing methods – clearly there are heuristic methods at work with that premise.

Eset NOD32 1.529

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

NOD32 produced an unsurprising 100 per cent detection rate and thus earns a VB 100% award. There was one fly in the ointment which affected many of the products in this review – this is by no means specific to *NOD32*, and neither was *NOD32* one of the worst offenders. The problem is that over 50 per cent of the products require a reboot when installing. Given that *Windows 2003 Server* is a server platform, this seems likely to irritate administrators no end. Most distressingly, three of the products on offer (*Vet*, *Dr.Web* and *VirusChaser*) required reboots when changing on-access configuration. This is frustrating enough when testing the products, but would be far more so on, for example, a company's main SQL server.

FRISK F-Prot Antivirus 3.14b

ItW Overall	100.00%	**Macro**	100.00%
ItW Overall (o/a)	100.00%	**Standard**	99.79%
ItW File	100.00%	**Polymorphic**	97.53%

F-Prot AntiVirus showed impressive detection and a lack of false positives. This was amply sufficient to be rewarded with a VB 100% award. It is worth mentioning here that *FRISK*'s *Linux* product will be undergoing a standalone review in the next issue. In the last *Linux* Comparative the on-access component proved intractable on our test machines and we hope to do the product more justice upon this occasion.

F-Secure Anti-Virus for Servers 5.41

ItW Overall	100.00%	**Macro**	100.00%
ItW Overall (o/a)	100.00%	**Standard**	99.98%
ItW File	100.00%	**Polymorphic**	99.92%

On-demand tests	ItW File		ItW Boot		ItW Overall		Macro		Polymorphic		Standard	
	Number missed	%	Number missed	%	Number missed	%	Number missed	%	Number missed	%	Number missed	%
AhnLab V3Net	0	100.00%	0	100.00%	0	100.00%	79	98.18%	9119	45.48%	313	85.42%
Alwil Avast!	1	99.56%	0	100.00%	1	99.58%	18	99.56%	153	91.21%	13	99.73%
CA eTrust Antivirus	0	100.00%	0	100.00%	0	100.00%	4	99.90%	1	99.89%	0	100.00%
CA Vet Anti-Virus	0	100.00%	0	100.00%	0	100.00%	0	100.00%	2	99.87%	2	99.90%
CAT QuickHeal	0	100.00%	0	100.00%	0	100.00%	101	97.54%	1078	92.89%	318	82.60%
DialogueScience Dr.Web	0	100.00%	0	100.00%	0	100.00%	0	100.00%	0	100.00%	0	100.00%
Eset NOD32	0	100.00%	0	100.00%	0	100.00%	0	100.00%	0	100.00%	0	100.00%
FRISK F-Prot Antivirus	0	100.00%	0	100.00%	0	100.00%	0	100.00%	3	97.53%	3	99.79%
F-Secure Anti-Virus	0	100.00%	0	100.00%	0	100.00%	0	100.00%	1	99.92%	1	99.98%
GDATA AntiVirusKit	0	100.00%	0	100.00%	0	100.00%	0	100.00%	1	99.92%	0	100.00%
Grisoft AVG	0	100.00%	0	100.00%	0	100.00%	20	99.51%	257	85.97%	23	99.01%
Kaspersky KAV	0	100.00%	0	100.00%	0	100.00%	0	100.00%	1	99.92%	0	100.00%
MicroWorld eScan	0	100.00%	0	100.00%	0	100.00%	0	100.00%	1	99.92%	0	100.00%
NAI McAfee VirusScan	0	100.00%	0	100.00%	0	100.00%	0	100.00%	0	100.00%	3	99.79%
Norman Virus Control	3	99.67%	0	100.00%	3	99.68%	3	99.93%	180	91.24%	4	99.87%
NWI VirusChaser	1	99.56%	0	100.00%	1	99.58%	4	99.90%	0	100.00%	0	100.00%
SOFTWIN BitDefender	0	100.00%	0	100.00%	0	100.00%	13	99.69%	23	96.50%	38	99.01%
Sophos Anti-Virus	0	100.00%	0	100.00%	0	100.00%	8	99.80%	18	98.06%	14	99.49%
Symantec SAV	0	100.00%	0	100.00%	0	100.00%	0	100.00%	0	100.00%	0	100.00%
Trend ServerProtect	0	100.00%	0	100.00%	0	100.00%	0	100.00%	515	94.38%	3	99.85%
VirusBuster VirusBuster	0	100.00%	0	100.00%	0	100.00%	0	100.00%	101	91.45%	8	99.82%

Without a review offer or interesting piece of gossip to fall back on, it would have helped if *F-Secure Anti-Virus* were to contain some glaring fault or bizarre easter-egg. Sadly for me this was not to be the case – the most that could be commented upon being a very slight slow down of the test machines after processing some rather vast log files. This persisted rather longer than might be expected but went away of its own accord.

After that earth-shattering revelation it will be no surprise that *FSAV* is a recipient of a VB 100% award for excellent detection and no false positives.

GDATA AntiVirusKit 12.0.5

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

AVK also showed excellent detection rates. However, the *GDATA* product managed to fall at the last hurdle – by throwing up a single false positive. The fact that this was produced by the *RAV* engine adds a little poignancy to proceedings. Not only does it show that the engine, although whisked away to Redmond, is still in use

commercially, but it also allows conspiracy theorists to blame *Microsoft* for *AVK*'s failure to gain a VB 100%.

Grisoft AVG Anti-Virus System 6.0.524 321

ItW Overall	100.00%	**Macro**	99.51%
ItW Overall (o/a)	100.00%	**Standard**	99.01%
ItW File	100.00%	**Polymorphic**	85.97%

The *AVG* product as supplied consists of a base package upon which updates are applied as a proprietary .BIN format. Rather oddly, however, some of the updates would operate only by being selected for use manually, while others were usable by insertion into the upgrades directory of the *AVG* installation. Also somewhat mysterious was the difference in detection between on-access and on-demand scans. It is possible that this was related to time-outs during on-access scanning of complex polymorphic samples, though that is not wholly convincing. What is certain, however, is that *AVG* gains a VB 100% award, with no false positives and full detection of ItW viruses both on access and on demand.

Kaspersky KAV 4.5.0.58

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

Installing a *Kaspersky AntiVirus* instance for the first time is something of a long-winded chore since numerous definition files must be layered one upon another before the product can be used. After installation the update process is rather easy and automated. As might be expected by this stage *KAV* detected well – the only misses being one of W32/Etap and the .VXD samples of Navrhar on-access, which were presumably avoided for reasons of scanning speed. Less predictable was the behaviour of on-access floppy scanning. Detection of disk changes was extremely variable and the only rational explanation for the times at which scanning occurred seemed to be that the on-access scanning was on a mini-schedule, ignoring disk changes and simply scanning at intervals. However, with no problems concerning detection or false positives, *KAV* earns a VB 100% award.

MicroWorld Software eScan 10,1,0,2

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

eScan opened with an impressive display of raw confusion – a server product which declared it could not be operated … on servers. The software was unusable as a result of a Flash animation of, among other things, an exploding planet, rendering the program no more than a rather large cartoon clip. Sadly matters became more mundane after this, and the program was able to be installed by means of ripping temporary installation files from their resting places. Full detection of all but one polymorphic file, and no false positives, led to a VB 100% award. A disappointment to me, since I was at least expecting to be assaulted by a horde of killer gannets.

NAI McAfee VirusScan 7.10 4.2.60 4296

ItW Overall	100.00%	**Macro**	100.00%
ItW Overall (o/a)	100.00%	**Standard**	99.79%
ItW File	100.00%	**Polymorphic**	100.00%

The testing of *McAfee VirusScan* did not bring back memories of the great floppy scandals of days gone by, and detection was impressive by any standards. I cannot remember a time when *VirusScan* produced a false positive in *VB* testing, and this review was no different. Another VB 100% is duly awarded.

Norman Virus Control 5.60.13

ItW Overall	99.68%	**Macro**	99.93%
ItW Overall (o/a)	99.68%	**Standard**	99.87%
ItW File	99.67%	**Polymorphic**	91.24%

To *Norman*'s chagrin the gremlins within the Sandbox technology at the heart of *NVC*'s heuristic capabilities have not yet sated their lust for fame. A false positive in the clean test set was accompanied by a warning of infection by Sandbox: W32/FileInfector. This will be irksome indeed for *Norman* but worse was to come. The on-access component of the program was highly unstable, tending to lock up the machine after being bombarded with a thousand or so viruses in short order. This problem made it impossible to perform testing in any but the ItW set, the other areas being too prone to cause the machine to enter a state of comatose narcissism. As if these woes were not enough to contend with, W95/Tenrobot.B was not fully detected in the wild.

NWI VirusChaser 5.0

ItW Overall	99.58%	**Macro**	99.90%
ItW Overall (o/a)	99.58%	**Standard**	100.00%
ItW File	99.56%	**Polymorphic**	100.00%

Hard Disk Scan Rate	Executables			OLE Files			Zipped Executables		Zipped OLE Files	
	Time (s)	Throughput (KB/s)	FPs [susp]	Time(s)	Throughput (KB/s)	FPs [susp]	Time (s)	Throughput (KB/s)	Time(s)	Throughput (KB/s)
AhnLab V3Net	23	23779.7		8	9916.7		46	3465.6	15	4973.8
Alwil Avast!	168	3255.5		16	4958.4		117	1362.5	30	2486.9
CA eTrust Antivirus	87	6286.6	2	4	19833.4		52	3065.7	8	9325.9
CA Vet Anti-Virus	78	7012.0		7	11333.4		49	3253.4	10	7460.7
CAT QuickHeal	60	9115.5		11	7212.2		45	3542.6	16	4663.0
DialogueScience Dr.Web	280	1953.3	[1]	14	5666.7		102	1562.9	18	4144.9
Eset NOD32	59	9270.0		7	11333.4		40	3985.4	5	14921.5
FRISK F-Prot Antivirus	89	6145.3		5	15866.8		57	2796.8	6	12434.6
F-Secure Anti-Virus	204	2681.0		9	8814.9		107	1489.9	18	4144.9
GDATA AntiVirusKit	425	1286.9	1	18	4407.4		210	759.1	30	2486.9
Grisoft AVG	126	4340.7	[6]	8	9916.7		50	3188.3	10	7460.7
Kaspersky KAV	126	4340.7		13	6102.6		82	1944.1	22	3391.2
MicroWorld eScan	140	3906.7		20	3966.7		81	1968.1	27	2763.2
NAI McAfee VirusScan	92	5944.9		15	5288.9		64	2490.9	20	3730.4
Norman Virus Control	68	8043.1	1	7	11333.4		63	2530.4	13	5739.0
NWI VirusChaser	168	3255.5	[12]	9	8814.9		61	2613.4	10	7460.7
SOFTWIN BitDefender	1576	347.0	[2]	8	9916.7		602	264.8	16	4663.0
Sophos Anti-Virus	59	9270.0		9	8814.9		44	3623.1	11	6782.5
Symantec SAV	126	4340.7		19	4175.5		61	2613.4	19	3926.7
Trend ServerProtect	57	9595.3		4	19833.4		28	5693.4	6	12434.6
VirusBuster VirusBuster	195	2804.8		7	11333.4		125	1275.3	16	4663.0

Being based on *Dr.Web*, *VirusChaser* shares the fussiness over changing on-access settings which characterises that program. Unfortunately *VirusChaser* also has fewer options available for the on-access scanner, thus making it more difficult to achieve the optimal settings for, among other things, testing. Since file-blocking on detection was not supported, in the end the collection was XCOPYed and logs used. Logs are not the preferred method for on-access scanning since, in my experience, they have a disturbing tendency to have missing entries. *VirusChaser*'s detection rate was distinctly sub-standard to that of the parent product. BAT/Mumu.A was missed both on access and on demand, and although no true false positives were seen, there were numerous suspicious files in the clean set.

SOFTWIN BitDefender Standard Edition 7.1

ItW Overall	100.00%	**Macro**	99.69%
ItW Overall (o/a)	100.00%	**Standard**	99.01%
ItW File	100.00%	**Polymorphic**	96.50%

With two suspicious files in the clean set, *BitDefender*'s start was not awful, but certainly ripe for improvement. Sure

enough detection rates were good, and the lack of any true false positives merited a VB 100% award. Once again *BitDefender* achieved the slowest scanning rates in the review, though this situation should not last long. The developers noted that, in their tests, scanning speed was far more reasonable and that some of *VB*'s files might be causing the problems. With the aid of a few logs it seems likely that *SOFTWIN* will be able to trim these times considerably.

Sophos Anti-Virus 3.74

ItW Overall	100.00%	**Macro**	99.80%
ItW Overall (o/a)	100.00%	**Standard**	99.49%
ItW File	100.00%	**Polymorphic**	98.06%

After having suffered in the polymorphics from the eternal affliction of ACG non-detection, *SAV* is now able to root out these infections in their entirety. Not only do I have fewer numbers to input as a result of this, but the folk at *Sophos* can also feel suitably relieved. Some things do not change however. All the criteria for a VB 100% award were easily achieved by *SAV*, and thus one wends its way to them through the figurative ether.

Symantec SAV Corporate Edition 8.1.0.25

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

Symantec AntiVirus's detection of all viruses in the test set, combined with the usual lack of false positives, earned *SAV* another VB 100% award. Scans of the on-demand test sets took over four hours, in comparison with four minutes as a more typical time for some products. This is really only a problem if you have a massive infection problem prior to installing *SAV*, though in this situation desperation to be uninfected might well overcome any care for speed. On the clean files no such speed problems were noted.

Trend Micro ServerProtect 5.56 Build (1007)

ItW Overall	100.00%	**Macro**	100.00%
ItW Overall (o/a)	100.00%	**Standard**	99.85%
ItW File	100.00%	**Polymorphic**	94.38%

ServerProtect was unique among the programs on offer in requiring to be set up in a domain with an active domain controller. It also needs to be set up with a distribution server, making it pre-configured for network updates and upgrades, but total overkill on a single machine. It also causes considerable issues when attempting to update with standard definitions rather than, directly or indirectly, through a net connection. Then again, anyone using a *Windows 2003 Web Edition Server* on its own has rather more problems to contend with already. It came as a great surprise that logging seemed constrained by the amount of data that could be processed at any one time. Due to the methods used to store the data, the information concerning a scan of 25,000 viruses seemed to take up rather more than 50 MB. This information was too great for the log parser to assimilate, thus logs were truncated when exported. As a result testing was performed by deletion. With these niggles out of the way, scanning was speedy and detection good, though slightly weak on polymorphics. With no false positives, *Trend*'s VB 100% award is well deserved.

VirusBuster VirusBuster 4.4 Build 2

ItW Overall	100.00%	**Macro**	100.00%
ItW Overall (o/a)	100.00%	**Standard**	99.82%
ItW File	100.00%	**Polymorphic**	91.45%

The final product in the review is again one which performs too well for any lovingly crafted words of vitriol to be appropriate. *VirusBuster* continues to show sturdy detection and with no false positives is due a VB 100%.

CONCLUSION

Overall the review held in store more surprises than I had expected; the problems relating to the new OS were fewer and the traditional problems were more profound than I thought likely. Although those companies who suffered may not find it particularly comforting, the pain of upgrading from one OS to another seems to be lessening overall. Not that this should be a concern for a couple of years yet – with the next generation of *Windows* looking increasingly likely to be delayed further, system patches rather than replacements will more than likely be the order of the day.

Technical details:

Test environment: Identical 1.6 GHz Intel Pentium machines with 512 MB RAM, 20 GB dual hard disks, DVD/CD-Rom and 3.5-inch floppy drive running *Windows Server 2003 Web Edition V5.2 Build 3790*.

Virus test sets: Complete listings of the test sets used are at http://www.virusbtn.com/Comparatives/Win2K/2003/test_sets.html.

END NOTES & NEWS

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

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

The Infosecurity.nl exhibition takes place 11–12 November 2003 at Jaarbeurs complex, Utrecht, Netherlands. For details of the show see http://www.infosecurity.nl/.

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

The Conference on CyberSecurity, Research, and Disclosure takes place at Stanford Law School, 22 November 2003. The conference will explore the relationship between computer security, privacy, and disclosure of information about security vulnerabilities. For details see http://cyberlaw.stanford.edu/conferences/.

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. Register before November 17 for a reduced conference registration fee. 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 are now being accepted for the Briefings and will be received and reviewed until 10 December 2003. Meanwhile, the call for papers for the Black Hat Europe Briefings (Amsterdam, Spring 2004) and for the Black Hat Briefings USA (Las Vegas, 26–29 July 2004) will open 15 November 2003 and 15 February 2004 respectively. 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/.

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/.

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

EDITOR'S NOTE

Welcome to the first edition of the *VB Spam Supplement*. Over the last 12 months *VB* has taken a couple of fleeting glances into the anti-spam arena and has decided that now the time has come to look into the issue in more depth. Spam is a subject that is of growing concern to the large proportion of the population who use email as a form of communication, and has proved to be of increasing interest to the AV industry: over the last year we have seen a veritable stampede of AV vendors rushing to bring anti-spam solutions to market alongside their anti-virus products.

With volumes of unsolicited email growing almost by the day, there is great interest in the ways in which users are affected by it. As a consequence, a rash of reports, surveys and questionnaires has appeared across the Internet. The results of these surveys suggest that spam is beginning to affect the way in which people use the Internet: 25 per cent of email users surveyed by *Pew Internet* said that their overall use of email had declined as a result of the increasing volume of spam. Of some concern was the fact that one survey (carried out by *DoubleClick* and *AOL)* reported that unsubscribing was one of the most common actions taken by respondents to limit spam – evidently the message has yet to filter through to home users that replying to or clicking on any links within unsolicited email will lead to an increase, not a decrease in spam. More pertinent to *VB* readers are the results of a survey of IT decision makers carried out by *Trend Micro* and *TechRepublic*. More than 50 per cent of respondents estimated that their organisations had experienced a 25–100 per cent increase in the volume of spam received over the past three months and approximately one third of respondents believed that viruses originate in the spam received by their organisations.

So spam is a hot topic, and deservedly so – as we hear from Martin Lee on p.S2, *MessageLabs* statistics indicate that the ratio of spam to non-spam messages exceeded the 50% mark earlier this year. *VB* plans to present a selection of news and articles on spam and anti-spam techniques – some technical, some ethical, some relating to real-world experiences of dealing with spam, and as time progresses we hope to look in depth at some of the anti-spam products on the market. As always, *VB* welcomes your comments, questions, suggestions and contributions – please email editor@virusbtn.com.

NEWS & EVENTS

UNANIMOUS VOTE FOR CAN SPAM ACT

The US Senate has approved the first federal anti-spam legislation. The Can Spam Act was approved by Senate in a unanimous 97-0 vote at the end of last month. The bill requires bulk commercial emailers to include both opt out provisions and valid email headers and subject lines in their solicitations; failure to do so will carry civil and criminal penalties. The bill provides for a maximum civil penalty of US$1.5 million for wilful violation of the law and up to five years imprisonment for the use of common spamming techniques. More controversially, the bill calls for the Federal Trade Commission (FTC) to create a 'Do-Not-Spam' list, along similar lines to its Do-Not-Call registry which it set up to shield consumers from unwanted telemarketing communications. However, the idea of a 'Do-Not-Spam' registry has not been welcomed by the FTC, since it does not believe such a list can be secured satisfactorily. With the bill having sailed through Senate, the focus now switches to the House of Representatives – encouragingly, a statement issued by the White House indicated that President Bush would sign an anti-spam bill.

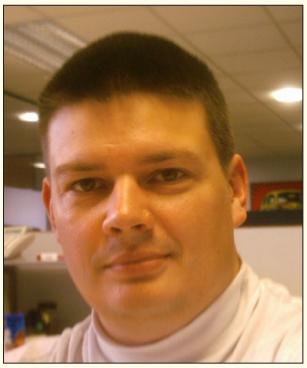
SPAM CONFERENCE

The 2004 Spam Conference will take place on 16 January 2004 at MIT, Cambridge MA, USA. The format is a series of quick, concentrated talks on new ideas and techniques for eliminating spam. There is no fee, but prior registration of attendees is compulsory. See http://spamconference.org/.

FEATURE

LEGAL ATTEMPTS TO REDUCE SPAM, A UK PERSPECTIVE

Martin G Lee
Anti-spam software engineer, UK



It is clear that the ever increasing deluge of spam is becoming a real nuisance. As such, it is to be applauded that governments throughout the world are taking notice and attempting to introduce a regulatory framework whereby legitimate email can be distinguished legally from the nuisance of unsolicited bulk commercial email, and those who insist on sending spam may be dealt with accordingly.

However, such an approach is fraught with difficulties. Poorly worded legislation risks legitimising spam, introducing loop holes that spammers can exploit – or, indeed, outlawing the legitimate practice of sending one-off emails to people you have never met.

This article summarises from a UK perspective the various legislative attempts to ban the abuse of email by law.

DATA PROTECTION

The unregulated and increasing processing of personal data, including email addresses, caused sufficient concern for the EU to pass the Data Protection Directive (95/46/EC) in the mid 1990s. This established that the processing and storage of personal information must be carried out with consent of the individual and with regard to the individual's rights to privacy.

The provisions of this directive were passed into UK law with the 1998 Data Protection Act. Nevertheless, this did not halt the collection and processing of email addresses by spammers. Presumably the posting of a personal email address on a web page or in a Usenet post was taken by the spammers as an indication of permission to process and store such information.

The EU Electronic Commerce Directive (2000/31/EC), which was integrated into UK law as the Electronic Commerce Regulations 2002, clearly states that '[the sender] shall ensure that any unsolicited commercial communication sent by him by electronic mail is clearly and unambiguously identifiable.'

This law renders all spam that attempts to masquerade as legitimate email illegal. So far, however, this appears to have had little effect – the spam keeps coming, mostly unmarked.

It is to be imagined that identifying a spam as such in the subject line is effective in reducing the number of recipients who open and respond to the email. Hence, the spammers prefer not to comply with the law – and in any case most spam is sent from countries outside of the EU where the senders do not feel obliged to follow EU law.

'Even in the absence of specific anti-spam laws, recipients and ISPs can seek to prevent spammers sending them spam and recover the costs involved in processing spam.'

PROSECUTION

Meanwhile in the US, existing laws were being used to combat the loss caused by processing spam and to prosecute fraudulent claims contained in spam.

AOL scored a major victory when it sought an injunction against *CN Productions Inc.* in 1998. The company objected to *CN Productions* sending spam to *AOL* subscribers, claiming that this was against *AOL*'s terms and conditions, that it cost *AOL* time and money to process the emails, and that the spoofing of the From headers to make it appear that the emails were coming from 'aol.com' was having an adverse effect on their reputation. The Virginia judge agreed and awarded *AOL* $1,819,863 in damages plus legal costs.

This case demonstrates that even in the absence of specific anti-spam laws, recipients and ISPs can seek to prevent spammers sending them spam and recover the costs involved in processing spam.

ACROSS BORDERS

Similarly, in 1999 a British provider of email services, *BiblioTech*, sought damages through the Georgia state courts in the US for the costs of processing the undeliverable message bounces generated by a spammer that were relayed to the company's servers.

Although Sam Khuri and his Atlanta print company *Benchmark Print Supply* tried to push for an out of court settlement, *BiblioTech* eventually won an undisclosed sum of damages and an injunction preventing Sam Khuri, the main defendant, from ever sending unsolicited bulk email. Thus, spammers can be pursued across national borders.

EVER-INCREASING CIRCLES

Nevertheless, despite these court rulings and increasingly strict legislation being introduced in the EU and across the US to govern unsolicited email, the volume of spam keeps increasing. In May 2003 the ratio of spam to non-spam emails passed the 50 per cent mark, according to *MessageLabs*' statistics – a 40.6 per cent increase over the preceding 12 months.

'In May 2003 the ratio of spam to non-spam emails passed the 50 per cent mark.'

A further tightening of the regulatory framework is due to be introduced in the Privacy and Electronic Regulations 2003, implementing EU directive 2002/58/EC. This law prevents the sending of unsolicited email 'unless the recipient of the electronic mail has previously notified the sender that he consents'. But will further regulation make any difference to the volume of spam?

Identifying spammers is not necessarily easy when emails are relayed through unsecured proxies or relays hiding their origin. Spam is a worldwide problem; emails can be sent from any country or jurisdiction to arrive in any other. The time, cost and sheer effort involved in tracking down and prosecuting the sender of an unsolicited message is prohibitive to all but the most tenacious or slighted companies and individuals.

To put the legal effort in context, one of the earliest and most well known legislative codices contains the law 'Thou shalt not steal', nevertheless some 3000 years after this was written theft continues to blight society. Despite the existence of laws and law enforcement assistance, the onus is on the individual to protect their possessions from theft through the use of good security and appropriate concealment.

It is likely to be a similar case for protecting the individual's inbox from spam. Invest in a good spam filter to prevent the spam from clogging your inbox, and be wary of broadcasting the existence of your most precious email addresses to people you do not trust completely.

Legislation assists in identifying clearly what is and what is not acceptable, but ultimately while there is money to be made through the sending of spam, this is not a problem that is going to go away any time soon.

Martin Lee is a software engineer in MessageLabs' anti-spam team writing in a personal capacity. The opinions and interpretations expressed here may not reflect those of his employer.

SUMMARY

ASRG SUMMARY

The Anti-Spam Research Group (ASRG) is one of a number of research groups that fall under the umbrella of the Internet Research Task Force (IRTF). The ASRG focuses on the problem of unwanted email messages, or spam, and its purpose is both to understand the problem and collectively to propose and evaluate solutions for the problem.

The ASRG is an open research group, whose meetings and mailing list are open to all participants. An archive of the mailing list is available at the ASRG mail archive (see http://www.irtf.org/asrg/), but *VB* intends to present a monthly summary of the postings to the mailing list which we hope will give you a flavour of the ongoing discussions without having to get bogged down with the minutiae of the to-ing and fro-ing of messages. This summary will also be available on the *VB* website – see http://www.virusbtn.com/.

ASRG OCTOBER 2003

This month the mailing list has been summarised by VB*'s Pete Sergeant.*

October's postings kicked off with David Nicol posting a link to an article in *eWeek* (http://www.eweek.com/), 'Should senders pay for the mess we call email?', and asking for comments on the piece. The main points of the article, as summarised by Yakov Shafranovich (and now resummarised by me) were that 'Sender Pays' would not work because:

- Since the Internet is global, the coordination of 'Sender Pays' would require international cooperation at a government level.
- 'Micro-payments remain a problem'.
- This could cause a problem for those who offer free email services.
- Hijacked machines could cause problems.

Several people politely disagreed with all the points listed above – the existence of international money transfer organisations, the existence of some micropayment solutions and so on, were mentioned in the rebuttals. Yakov's plea for a summary sadly went unanswered.

Brett Watson defended the use of 'Pull' techniques where, in the words of Dan Bernstein (DJB) 'the *sender*'s ISP, rather than the receiver's ISP, is the always-online post office from which the receiver picks up the message'. Brad Knowles was unconvinced, and a series of exchanges between the two followed. Those interested in the rest of DJB's succinct writings on the subject could point their browsers to http://cr.yp.to/im2000.html.

Peter Kay commented on Eric Dean and Yakov's draft Challenge/Response Interworking (CRI) proposal, and was unhappy about the recommendation that challenge-response systems should send challenges from a user other than the intended recipient, pointing out that many challenge-response systems will white-list recipients of outgoing mail, thus avoiding the need for CRI overhead in those cases. A long discussion followed, including a tale of woe concerning Challenge/Response Hell caused by bug ticketing systems.

Yakov announced the creation of a mailing list for dialogue between the authors of SPF, RMX, DMP, and other designated sender schemes, with Alan DeKok as coordinator. The SPF website (http://spf.pobox.com/) claimed that a draft specification was almost ready, and showed how SPF would look using it, but warned that implementers should wait for version two of the draft.

Yakov published a link to Curtis M. Kularski's draft on 'Creative Addressing' – the general response to which seemed to be that there were a good number of questions unanswered, or that the draft added very little new content. Curtis responded by posting links to earlier versions of his draft that didn't 'lack beef', saying he'd had to change the draft significantly in order to try to get it accepted by the RFC Editor. The RFC Editor has received that draft and passed it on to the ASRG for commentary. [As a side note, *Yahoo* announced in October that it would be implementing something similar, for paying users of its webmail service.]

Markus Stumpf came up with an interesting idea about spam taxonomy, which would allow people to refer very quickly to the different sorts of email they term spam, to facilitate communication. Sadly the posting received less of a response than it seemed to merit. Markus proposed four main headings under which a spam could fall. These were: 'Private mail', 'Targeted non-bulk mail' (such as contact with existing customers), 'Bulk email' (to include discussion lists and so on) and 'Automated messages' (such as bounce messages).

Andrew Akehurst posted a similar idea a couple of days later, which met with a favourable response, and he followed it up a little while later with the first draft of his email use-cases. Andrew summed up the major differences between his and Markus's ideas: 'My main criterion for the classification was to classify things into different categories only if a machine could reasonably recognize the difference between them,' and said he was sceptical about the chances of a machine being able to make the distinction between some of Markus's classifications.

Terry Sullivan talked about maintaining collections of spam for analysis, and the problems faced by those using spam-trap addresses – otherwise identical spam-trap addresses receive vastly different amounts of spam for no discernable reason. He suggested that a concerted effort was called for, which would either involve asking large organisations/ISPs for access to their spam-traps, or for the ASRG to set up its own spam-trap effort. He summed up by warning that, unless a concerted effort is made towards trap address maintenance, the only data that will be available for research will be (presumably out-of-date) archived data – a separate analysis group has been set up.

Kee Hinckley offered an explanation for the variance of spam volume between different addresses, pointing out that it is all too easy to forget that almost all spam is sent by about 200 major spammers, using a smaller number of varieties of mailing software, to an even smaller number of address sources. He went on to say, 'What we are sampling is not spam, but spammer targets/techniques', and agreed that 'accurately measuring such a small population may require a much greater distribution of spam traps'. Yakov asked Terry to clarify what exactly he wanted, pointing out that there are several sources for archived spam. Terry replied by saying that he was mainly trying to solicit feedback, and Kurt Magnusson spelled out why he thought access to a near real-time source of spam was more useful than dated archives.

Andreas Saurwein stated that, in his opinion, 15 per cent or so of the pieces of spam found on http://spamarchive.org/ are not really spam, and that this represents a hindrance to running analysis tools against the corpus.

Paul Judge made the point that people have different ideas of what constitutes spam, and suggested a couple of ideas that had been bandied about for refining and filtering the corpus, including the use of anti-spam products and some form of 'Am I Spam Or Not?' voting system. [Some readers may find Vesselin Bontchev's paper on maintaining a virus library provides an interesting parallel – see http://www.virusbtn.com/old/OtherPapers/VirLib/.]

Kurt Magnusson was mildly surprised to find that he had stopped receiving spam from Korea and asked if anyone had noticed anything similar. Apparently no one else had observed any decline in spam from that part of the world.

Yakov pointed out the existence of the Best Current Practices list, of which Brad Knowles is the coordinator. Brad posted some information about this non-advertised list to the ASRG list.

Finally, this month's award for saying an awful lot without actually saying a great deal goes to the person who wrote the following: 'Our unique multifaceted approach to solving Spam, which has been in production since early 2002, is based on a patent pending DNA-like sequencing technology which is languages independent, highly reliable, accurate, and extremely secure.'