JANUARY 2006

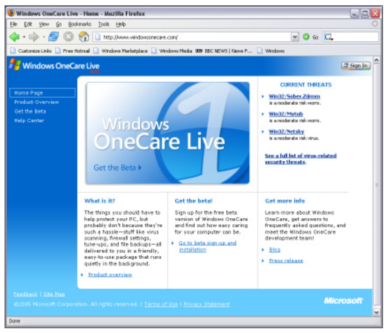
virus BULLETIN

The International Publication on Computer Virus Prevention, Recognition and Removal

CONTENTS

ISSN 1749-7027

IN THIS ISSUE



Win32/Sober.Z@mm is a moderate risk worm.
Win32/Mytob is a moderate risk worm.
Win32/Netsky is a moderate risk virus.
See a full list of virus-related security threats.

What is it?
The things you should have to help protect your PC, but probably don't because they're such a hassle—stuff like virus scanning, firewall settings, tune-ups, and file backups—all delivered to you in a friendly, easy-to-use package that runs quietly in the background.
Product overview

Get the beta!
Sign up for the free beta version of Windows OneCare and find out how easy caring for your computer can be.
Go to beta sign-up and installation

Get more info
Learn more about Windows OneCare, get answers to frequently asked questions, and meet the Windows OneCare development team!
Blog
Press release

Feedback | Site Map
©2005 Microsoft Corporation. All rights reserved. | Terms of Use | Privacy Statement

BETA ON TEST

VB usually shies away from the testing of beta products, but the beta release of *Microsoft*'s anti-virus offering *OneCare Live* proved to be just too tempting. Matt Ham describes his findings when he put *OneCare Live* under the microscope.

page 16

INFORMATIONAL NEEDS

Information is a vital piece in the armour for corporate security analysts defending their organisations against the latest malware threats. *VB* advisory board member Jeannette Jarvis explains why it is so important for customers and vendors to understand each other's information needs and work together to provide solutions.

page 2

vbSpam supplement

This month: anti-spam news & events, and Gordon Cormack reports on the results of the TREC 2005 spam filter evaluation effort.

COMMENT

'Information is ... our lifeblood for details about computer threats, impact and activity.'

Jeannette Jarvis
The Boeing Company, USA

ALLY IN OUR DEFENCES

My organization is responsible for the protection of the computing infrastructure of *The Boeing Company*. This can be a monumental task.

Information is a strong ally in our defences. It is our lifeblood for details about computer threats, impact and activity. The sooner we know of any new threat and its exact payload, the sooner we can implement mitigation strategies. We already employ best practices to keep malware threats outside our company, but we back that up with additional countermeasures. We promote a defence-in-depth strategy that ensures multiple detection points for any new threat. This approach has proved beneficial, particularly on those rare occasions when a product fails to do its job, or when a security vendor does not get timely updates to us.

We monitor numerous anti-virus vendor websites, security vendor websites and alert streams, and other forums literally every hour of every day, screening for new information about computer threats. This constant search for new information allows us to implement protection measures even when the vendors we use are not publishing information or do not have detection available yet.

Our extensive monitoring has shown that many discrepancies exist between vendor write-ups. All security vendors must have current and correct information regarding all viruses listed on their web sites. There is always a concern that if the exploit information is not correct or complete, then the detection may not be correct or complete either. The integrity of the information you publish reflects the integrity of your products.

Sometimes a vendor will not publish new exploit information until an update is available. This is a disservice to customers who may be able to use that information to implement blocking measures to keep the threat out until the vendor can provide the detection updates. A top concern for us is getting information about exploits targeting vulnerabilities in products or operating systems that may not have patches available.

Two details we find valuable that are often missing from virus information are alias names and timestamps that reflect data changes. Providing alias names on all threats would allow the group that provides our monitoring service to correlate the information amongst vendors more easily. We are not suggesting that vendors provide every single alias name available, but provide at least a fair sampling. Of course having a Common Malware Enumeration (CME-ID) identifier for all threats would be the optimum situation. When vendors use a timestamp to reflect changes to their write-ups, we can peruse their sites more easily. Because we seek so much information, we need to be able to find new information quickly, without having to re-read the original details.

Some security providers seem apprehensive about sharing the complete details of threat propagation with corporate customers. I understand concerns regarding publishing links that give access to downloadable malware. For that reason, I advocate creation of two information streams: one for the general public, which does not include the entire malicious URL, and another for your corporate security analysts, who can handle that information correctly.

Some excellent sources of information have been instrumental in getting new threat information into our hands quickly. AVIEN and AVIEWS are grass roots forums that address information sharing. Both forums have given us critical and timely intelligence around exploits.

AVIEN and AVIEWS have also helped build collaboration between customers and security vendors. We really can do more together than we ever can alone. Just as these forums took information sharing to a new level, they are expanding the possibilities yet again with the inaugural webcast conference on January 18. This webcast is one more example of customer-led change.

We all need to continue to understand each other's information needs and work together to provide solutions. As Henry Ford stated 'Coming together is a beginning. Keeping together is progress. Working together is success.'

Editor: Helen Martin

Technical Consultant: Matt Ham

Technical Editor: Morton Swimmer

Consulting Editors:
Nick FitzGerald, *Independent consultant, NZ*
Ian Whalley, *IBM Research, USA*
Richard Ford, *Florida Institute of Technology, USA*
Edward Wilding, *Data Genetics, UK*

NEWS

CSIA SETS AGENDA FOR US GOVERNMENT ACTION

The Cyber Security Industry Alliance (CSIA) has called for the US Federal Government to take action on a series of recommendations to strengthen the defences of the nation's information infrastructure in 2006.

The CSIA, an industry group which aims to ensure the privacy, reliability and integrity of information systems through public policy, technology, education and awareness, has produced a 'National Agenda for Information Security in 2006'. The Agenda identifies specific actions required to improve information security in the US. In addition, the Alliance has produced a report on the US government's progress (or what it considers the lack thereof) in strengthening information security over the past year. According to the report, 65 per cent of Americans believe that the government needs to make information security a higher priority than it currently is.

The CSIA calls on the US Administration and Congress to implement the following actions:

- Pass a national data breach notification bill.
- Pass a national spyware protection bill.
- Ensure cyber security protection is applied to the health care infrastructure.
- Promote information security governance in the private sector.
- Direct a federal agency to track the costs associated with cyber attacks.
- Secure digital control systems.
- Improve the quality of software security by strengthening NIAP certification.
- Fill new cyber security posts in the Department of Homeland Security.
- Ratify the Council of Europe's Convention on Cybercrime.
- Increase R&D funding for information security.
- Complete the HSPD-12 initiative for government-wide authentication.
- Ensure continuity of government operations with telework.
- Include information security planning in transition to IPv6.

Paul Kurtz, executive director of the CSIA, explained: 'We urge the government to take action on the 13 critical steps ... that we believe will help to immediately strengthen our information systems and begin to raise the confidence of our citizens in our networks.'

Prevalence Table – November 2005

Virus	Type	Incidents	Reports
Win32/Sober	File	4,199,499	93.16%
Win32/Mytob	File	164,426	3.65%
Win32/Netsky	File	106,559	2.36%
Win32/Mydoom	File	16,926	0.38%
Win32/Bagle	File	10,752	0.24%
Win32/Lovgate	File	3,821	0.08%
Win32/Funlove	File	1,734	0.04%
Win32/Zafi	File	751	0.02%
Win32/Sdbot	File	536	0.01%
Win32/Bugbear	File	504	0.01%
Win32/Mabutu	File	491	0.01%
Win32/Bagz	File	294	0.01%
Win32/Elkern	File	269	0.01%
Win32/Gibe	File	162	0.00%
Win32/Pate	File	146	0.00%
Win32/Dumaru	File	136	0.00%
Win32/Klez	File	130	0.00%
Win32/Chir	File	129	0.00%
Win32/Mimail	File	119	0.00%
Win32/Reatle	File	100	0.00%
Win32/Valla	File	80	0.00%
Win32/Maslan	File	51	0.00%
Win32/Bobax	File	40	0.00%
Win32/Gael	File	37	0.00%
Win32/Bofra	File	33	0.00%
Marker	Macro	31	0.00%
Win32/MyWife	File	17	0.00%
Win95/Tenrobot	File	17	0.00%
Win32/Monikey	File	11	0.00%
Win32/Swen	File	10	0.00%
Win32/Kriz	File	9	0.00%
Roor	Script	7	0.00%
Others[1]		66	0.00%
Total		4,507,893	100%

[1]The Prevalence Table includes a total of 66 reports across 26 further viruses. Readers are reminded that a complete listing is posted at http://www.virusbtn.com/Prevalence/.

FEATURE 1

INSIDE THE MICROSOFT SCRIPT ENCODER

Peter Ferrie
Symantec Security Response, USA

When the *Microsoft Script Encoder* was released in 1999, it was predicted that malware authors would use it to obfuscate their code. As a result, tools claiming to be able to decode the files produced by the script encoder started to appear almost instantly.

YOU BREAK IT, YOU BUY IT

Recently, I was given an encoded script to examine, and I was told that it might contain an exploit of some kind. Since I am not in the habit of carrying script decoding tools with me, I downloaded a script decoder tool and used it to decode the file.

The result was a partially decoded file containing a fragment of what looked like shellcode and a lot of garbage bytes. Since the file was in ANSI format, there were three possibilities: that the file contained double-byte character set (DBCS) characters which were not being decoded correctly on the English system; that the script was broken; or that the tool contained a bug.

Assuming the first case, I tried decoding the file using other language formats that the tool supported, but again I was unsuccessful.

There are four languages that use DBCS characters: traditional Chinese, simplified Chinese, Japanese and Korean. Since the tool that I had downloaded supported only three of these languages, I decided to download several other script decoder tools in the hope that one of them would support the remaining language.

When the half-dozen or so tools that I had downloaded also failed to decode the script, I ruled out the third possibility. How likely is it that every copy of a tool would have the same bug? (As a matter of fact this is more likely than one might imagine, as I found out afterwards via a completely unrelated matter: try searching for tools that enable 'Unreal' mode on x86 and x86-64 processors, notice how many people claim to have found it, and notice that none of them enable the A20 line first.)

That left the second possibility – that the script was broken.

SEE SCRIPT RUN, RUN SCRIPT RUN

Although I knew that running the script wouldn't provide a conclusive result, I ran it anyway. Sure enough, the Windows Scripting Host reported that the script was invalid. The question was: why?

Since I had long forgotten the details of encoded scripts, I downloaded the *Microsoft Script Encoder* tool (screnc.exe) and started to reverse-engineer it. Under normal circumstances, one would assume that examination of an encoder would be sufficient to provide an understanding of decoding methods. However, in this instance that is not the case, and it seems that the creators of the decoding tools all made the same mistake.

SCRENC.EXE

The first interesting thing I noticed about the *Microsoft Script Encoder* is that it supports ANSI, UTF-8, Little-Endian and Big-Endian Unicode input file formats. This is interesting because neither *Microsoft*'s own VBScript and JScript scripting engines (before encoding), nor the decoder built into those scripting engines (after encoding), support anything but ANSI and Little-Endian Unicode. Upon attempting to execute files in the other formats, the *Windows Scripting Host* reports that they are invalid.

The second interesting thing is that the encoding is done by the VBScript and JScript scripting engines themselves. The reason for this is that these engines support dynamic encryption, using the EncodeScriptFile method. It seems that this method was not noticed by malware authors.

The rest was fairly straightforward: encoded files begin with the signature '#@~^', followed by the base64-encoded length of the script that follows immediately. After the script is the base64-encoded checksum, and the signature '^#~@'.

The checksum is simply the sum of all of the characters from the script before it was encoded. It is used during the decoding phase to verify that the script has been decoded correctly, rather than to verify that the encoded script has not been altered.

If the script is not already in the Unicode format, it is converted to the Unicode format in memory prior to encoding. If the original file was in the ANSI format, the current code page is used to perform the translation, which causes the DBCS problem described above.

Once in the Unicode format, characters are not encoded if more than seven bits are required to identify them. In those cases, the character is simply copied instead. However, if the resulting encoded script is then saved in the ANSI format, any characters that cannot be represented in seven bits or fewer will be replaced by the system default for untranslatable characters (which is usually the '?'

character). If such a replacement is made, the script can no longer be decoded properly.

COMPOUND INTEREST

Now the fun begins – watching how *Microsoft*'s VBScript and JScript scripting engines deal with encoded files. Immediately we see that what screnc.exe produces is not all that those engines will accept.

The first thing the script decoder does is to search the entire script for the signature '#@~^'. This means that the script is not required to appear at the start of the file. Screnc.exe can produce such files only when the script is inside an HTML file. However, all of the decoding tools that I tried supported this behaviour.

The encoded script is decoded into the same location in the script at which it is found (but not the same location in memory). This means that unencoded script can appear before and/or after encoded script, even though screnc.exe cannot produce such files. None of the tools that I tried were affected by this, since they all found the script no matter where it was. However, one of the tools did not append the unencoded script that appeared after the encoded script.

The entire script is searched for all signatures that exist. This means that multiple encrypted scripts can appear in a single file! Screnc.exe cannot produce such files, and none of the tools that I tried supported it, either. But wait – it gets worse ...

WHAT THE #@~^?!

Any part of a script can be encoded, even down to the level of individual characters. The result is that a script such as:

```
oh_this="bad"
```

can become (unencoded characters are marked in bold):

oh_th#@~^AQAAAA==raQAAAA==^#~@**s="b**#@~^AQAAAA==CYQAAAA==^#~@**d"**

Fortunately, recursive encryption is not allowed, since the script decoder makes only a single pass over the script and decodes it to a different location in memory. If the script decoder had decoded to the same memory location, then it might have been possible to support recursive encoding to arbitrary levels, which would have made the problem much worse.

Given that an encoded script can appear anywhere in a file, it might seem surprising that all of the authors of the decoder tools made the same assumption: that the '#@~^' signature is a guarantee that what follows is an encoded script.

Of course, that's simply not true. Thus, the line:

```
x="#@~^" :#@~^AwAAAA==a{F5gAAAA==^#~@:msgbox(x)
```

was not decoded by any of those tools, yet *Microsoft*'s VBScript and JScript scripting engines decoded and executed it correctly (it prints '1', not '#@~^'). Also note the space that appears after the ' " '. Without it, even *Microsoft*'s VBScript and JScript scripting engines are fooled into believing that what follows the first '#@~^' signature is an encoded script, since the ':' character is a valid entry in the base64 dictionary that is used to decode the script length. The decoded length is an enormous value, and too large for the *Windows Scripting Host*, which reports that the script is invalid and exits without executing any further script.

Despite the fact that unencoded script can appear both before and after encoded script, the decoding is done before any script is interpreted, so string concatenation does not work. For example,

```
a="#@~^AQAAAA=="+"qMQAAAA==^#~@"
```

does not decode to 'a=1'. Additionally, decoding is not done after any script is interpreted, so this line:

```
eval(x)
```

where x is the encoded script

```
'#@~^AQAAAA==CYQAAAA==^#~@'
```

that was read from a file, is not decoded and does not evaluate to 'a'.

(THAT'S A BUG)

As mentioned above, the length and checksum of the decoded script are stored in base64 format. A bug exists in the base64 decoder in *Microsoft*'s VBScript and JScript scripting engines, which does not limit the input values correctly. This can be used to obfuscate the true length and/or checksum from tools which accept only files whose length and checksum are correct.

There is also an integer overflow bug in *Microsoft*'s VBScript and JScript scripting engines that causes a crash while calculating the length of the script. The bug is triggered if the top bit is set in a decoded length that would otherwise point within the file.

CONCLUSION

So what happened to that script? In the end, it was simply broken. There were extra characters inserted throughout the script, so the decoded length did not match; and there were some characters whose value was incorrect ('.' instead of the tab character, for example), so the decoded checksum did not match. After I had identified and removed the extra characters, and corrected the incorrect characters, the script decoded properly on an English system. It even contained an exploit, but it was one that we knew about already.

FEATURE 2

COOPERATIVE HEURISTICS

John D. Park
Symantec Security Response, USA

Non-heuristic anti-virus scanning is designed to be strict – it involves searching for matching checksums or specific bit patterns, which results in a very low false positive rate. The downside of strict scanning is that, for every new malware or new variant, a human malware analyst has to provide the scanner with a signature. Even with signatures, the scanner does not know what the malware looks like as a whole, what it does, or why it is malicious. In order to make scanning more flexible and proactive, I will teach an anti-virus scanner to think more like a human analyst.

Before going any further, here are the assumptions I have made:

1. Today's main threats are simple Trojan horses and worms, not file-infectors.
2. Malware is becoming more open-sourced, and construction-kit generated. Thus, much of the source code stays the same.
3. Source code can be compiled using different compiler options, and packed with various packers, which will result in different binaries.

In addition:

- Non-heuristic anti-virus scanning (checksum, string, p-code) is still necessary to detect file-infectors, and unique or more complex malware.
- A flawless emulation or memory dump is assumed.
- I have focused mainly on readable ASCII and Unicode strings. Strings contained in most of the prevalent threats have distinctive statistical characteristics such that readable strings provide enough information to identify them.

STRATEGY OVERVIEW

I used a number of different loose heuristics, and made them work cooperatively to verify each other, to reduce the false positive rate [1]. The three scanning methods used are: weighted mini-signature, statistical analysis using knowledgebase, and abnormal functionality detection.

SCAN #1 – WEIGHTED MINI-SIGNATURE

One of the most common non-heuristic anti-virus scanning methods is string matching. If we want to detect a malware called 'fruit basket', containing the string 'apple, banana, orange in a basket', we might add a string signature like 'ana, orange in a baske'. This means the scanner will detect 'apple, banana, orange in a basket', but it will miss variants like 'apple, banana, orange in a *yellow* basket', or 'banana, orange, *apple* in a basket'.

Semantically, a fruit basket is any basket containing any fruit. A more human pattern matching would use multiple short string signatures, or 'mini-signatures'. A mini-signature is shorter than a usual signature, and is too short and too generic to be used by itself. However, when used together with other mini-signatures, the risk of false positives is reduced.

My mini-signatures for 'fruit basket' would be 'apple', 'banana', 'orange' and 'basket'. A comma, a space, and the words 'in' and 'a' are omitted from the strings since they are not unique to 'fruit basket'. We can say that matching three out of four is good enough for detection, but it is not sufficiently flexible. So, a weight is added to each mini-signature:

Object	**Weight**	**Mini-signature**
fruitbasket	0.25	apple
fruitbasket	0.25	banana
fruitbasket	0.25	orange
fruitbasket	0.50	basket

The weighted detections by each mini-signature are added together and if the sum of the detected weights is more than or equal to 1, the detection of 'fruit basket' is triggered.

A mini-signature scanner would detect the following as 'fruit basket':

apple, banana, orange in a basket =
0.25 + 0.25 + 0.25 + 0.50 = 1.25

apple, banana, orange in a yellow basket =
0.25 + 0.25 + 0.25 + 0.50 = 1.25

banana, orange, apple in a basket =
0.25 + 0.25 + 0.25 + 0.50 = 1.25

basket, basket, basket in a basket =
0.50 + 0.50 + 0.50 + 0.50 = 2.00

apple, apple, orange, banana, apple =
0.25 + 0.25 + 0.25 + 0.25 + 0.25 = 1.25

And, it would *not* detect the following:

apple, apple, apple = 0.25 + 0.25 + 0.25 = 0.75

orange in a basket = 0.25 + 0.50 = 0.75

As we can see, this method is somewhat loose, and sometimes it is incorrect. But it is more flexible, like human recognition, than strict string matching.

The scanning method can be used with existing string databases seamlessly, where the weights to the current signatures would be 1.00.

CERTAINTY RATING

Since we have a weight for each mini-signature, we can do a calculation to get a certainty rating, ranging from 0% to 100%. The formula I used is:

1 - ([1 - DetectedWeight1] * [1 - DetectedWeight2] * .. * [1 - DetectedWeightN])

So the following examples would get certainty ratings of:

apple, banana, orange in a basket
1 - ([1 - 0.25] * [1 - 0.25] * [1 - 0.25] * [1 - 0.50]) = 78.9%

apple, banana, orange, apple in a basket
1 - (0.75 * 0.75 * 0.75 * 0.75 * 0.50) = 84.2%

banana, orange in a basket
1 - (0.75 * 0.75 * 0.50) = 71.9%

basket, basket, basket in a basket
1 - (0.50 * 0.50 * 0.50 * 0.50) = 93.8%

And, if these were to be detected:

foo, bar, baz
1 - (1 * 1 * 1) = 0.0%

apple, apple, apple
1 - (0.75 * 0.75 * 0.75) = 57.8%

As we can see, the certainty rating is not always accurate, but it is generally helpful in determining how certain the scanning result is.

Now, a malware scanner does *not* always have to give a binary result, like either 100% malware A, or 100% clean. It can give 95% malware B, or 24% malware C, or even 98% malware D and 70% malware E (where it's a cross-bred malware, such as W32/Mytob). With certainty ratings, users can set different thresholds – so, for example, government and financial institutions can have stricter scanning than regular home users.

I have crafted each of my mini-signature strings and their weights manually, covering more than 250 malware families, where each family contains 1 to 20+ mini-signatures. Scanning is case-insensitive to detect trivial variants.

SCAN #2 – MALWARE KNOWLEDGEBASE

This scanning method relies mainly on statistics. Every anti-virus company has its own collection of malware. These are very valuable resources, and many companies trade and share them, mainly to expand their signature databases to cover all known malware. I believe these resources have not been fully utilized, and more valuable data could be extracted from these collections.

Theoretically, by training neural networks with a sample set, a scanner should be able to distinguish malware from clean files. Work has already been done in this area by constructing neural networks of small sequences of bytes, called n-grams [1]. I took a different approach from n-grams and neural networks, mainly to reduce noise. Instead of looking for n-grams, I looked only for meaningful strings, such as filenames, IP addresses, email addresses, CLSIDs and URLs, using regular expressions. These strings, or 'components' are special in that a program communicates with the system through these strings. I compiled the components from known malware and known clean files into a malware knowledgebase, and labelled each entry with either the malware family name or 'clean'.

This is similar to the CLSID database from castlecops.com/CLSID.html, but extended to include other components. Since the regular expressions provided enough uniqueness for each string, I could get usable results without using neural networks or weights.

The rationale behind this scanning is as follows: malware A is known to use filenames 'Expl0re.exe', 'n0tepad.exe', 'scvhost.exe' and 'www.gogle.com', and no other files have been known to contain any of them. If you see an unknown file that contains all four of them, how likely is it that this file is malicious?

It can even work for clean components. An unknown file contains 80 'clean' components, like 'hi.txt', 'hello.txt', 'foo.txt' and 'bar.txt'. Of those 80 components, 60 are known to be related to malware B. What is the chance that two unrelated programs share 60 out of 80 components? Even if it is a pure coincidence that they share 60 components, it would be useful for the malware analyst to know the correlation. To enhance signal-to-noise ratio, some components, such as kernel32.dll and advapi32.dll, which occur in every program, are filtered out manually.

The promising feature of this malware knowledgebase scanning is that a known sample set or a trusted anti-virus scan can populate the malware knowledgebase with found components and their corresponding malware family names. Thus, the larger the knowledgebase grows, the more knowledgeable the scanner becomes.

Other applied uses of this knowledgebase are:

1. To find out whether a specific bank URL has been present in any known password stealers, so the bank can be warned.
2. To obtain a list of all URLs in all known downloaders, so that these downloader URLs can be monitored.
3. To obtain email addresses or URLs associated with certain password stealers, so that the server administrator can be asked to shut those down.
4. To find a difference between a new variant and all other known variants with respect to components.

Basically, the knowledgebase provides a single file view of all previously scanned malware and clean files.

SCAN #3 – ABNORMAL FUNCTIONALITY COMBINATIONS

It is perfectly legal for a law-abiding adult to drink alcohol. It is perfectly legal for a valid driver's licence-bearing adult to drive a vehicle. But, drinking and driving don't mix. The same logic applies to programs.

If a program contains a string 'SOFTWARE\Microsoft\Windows\CurrentVersion\Run' and 'advapi32.dll', it is likely that it has the capability to add an autorun registry key to start itself. If a program contains 'MIME-Version: 1.0', it is likely that it has the capability to send an email. If this program contains 'bank.com', it is likely that it has the capability to monitor online banking URLs.

It is fine for a program to add an autorun registry key. It is fine for a program to send out an email. It is fine for a program to contain a bank URL. But, it is *not* fine for a program to have an autorun registry key, a bank name, and an email send feature. This program is likely to be a password stealer for online banking that sends sniffed traffic via email. There are combinations of functionalities that just don't make sense in benign code.

This method is similar to mini-signature scanning in that each functionality is detected by adding up the detected weights. Monitored functionalities are: autorun registry keys, disable *Windows* tools, *Internet Explorer* start page, *Internet Explorer* zone settings, ID and password lists, process monitoring, downloaders, shell, sockets, anti-virus process lists, exploits, mass-mailers, email, *MSN Messenger*, *AIM*, IRC backdoor command list, P2P lure filenames, banks, Visual Basic, Borland Delphi, Visual C, JavaScript, diallers, generic adware and game registry keys. As you can see, none of the functionality is malware family-specific.

After each functionality has been identified, the results are compared to known illegal or dangerous combinations. The following categories are alerted in descending order of importance: IRC bot, generic backdoor, password stealer, *Internet Explorer* security setting changer, *Internet Explorer* start page hijacker, downloader, anti-virus process killer. Other than these general illegal combinations, there are other uses for this method, such as looking for out-of-place functionalities. For example, a CD key crack program downloaded from a warez site, should not be using sockets or contain any bank URLs. And, it would be odd for a genuine *Microsoft* file to be compiled using Borland Delphi.

Sometimes it can be hard to distinguish malware from a clean file just by looking at functionalities. For instance, an FTP server has similar functionalities to a backdoor: it listens for a command and sends or receives a file.

A NEW NAMING SCHEME

In order to link the results from the functionality scanner to a specific family, I needed a family name to show its functionalities. I needed the scanner to know that W32/Spybot spreads using exploits, has an IRC backdoor, downloads files, steals game keys, has embedded files, kills anti-virus processes and is written in C. So I changed the name W32/Spybot to 'WBspybotCDGMV'. The family name is in lower case, and functionalities are in upper case, where each character represents a known functionality. The preceding characters are 'necessary', and the following characters are 'optional'.

'WBspybotCDGMV' means that a W32/Spybot must be a worm (W) and an IRC backdoor (B), and is also likely to be written in C (C), have downloading capability (D), look for game keys (G), have multiple embedded files (M) and kill anti-virus processes (V). Other functionalities are: keylogging (K), *IE* startpage change (S), *IE* zone settings (L), *Windows* tool disable (R), generic adware (A), generic dialler (P), written in Delphi (E) and written in Visual Basic (Z). This is similar to Bezrukov's naming scheme [2], but I decided to keep the family name so that it can be referenced with an easy name. I removed all dots, slashes and dashes since they cause unnecessary naming differences among vendors.

The 'optional' fields are implemented to accommodate leaner variants, which lack non-essential functionalities, yet are categorized under the same family since they share the major functionalities. The naming scheme not only reduces analysts' work from reading existing malware descriptions, but could also be used to generate a preliminary malware description on-the-fly from pre-defined templates with data fed from the knowledgebase scanner.

NUTRITION FACTS FOR SOFTWARE

This naming scheme is not exclusive to malware, and can be applied to software in general. The software industry could adopt something similar to the Nutritional Facts label found on packaged food. Every food approved by the US Food and Drug Administration (FDA) has an easy-to-read label, giving a break down of the nutritional content of the food, such as '47g of sugar', '45mg of sodium', and so on. 'Nutritional facts for software' could display whether the software has the following functionalities: outbound connection, inbound connection, email, no GUI, autorun registry keys, driver, registry keys, pop-up windows, monitors other processes, update, dialler, etc. This would give users a better idea of what the program might do to the system, and would be a lot easier to read than the End User License Agreement (EULA).

If this practice were to become widespread, security products could detect unlabelled functionalities. Furthermore, as a preventative measure, more advanced OSs could authorize only labelled rights to the program, while denying any unlabelled rights – a simpler and cleaner task than anti-virus scanning.

THREE STRIKES, YOU'RE OUT

Let's say we scan an unknown file with all three scanners. The mini-signature scanner says the file is W32/Spybot with a weights sum of 7.3 and a certainty rating of 95.3%. The knowledgebase scanner tells us that 25 out of 28 components match with previously known components related to W32/Spybot. Finally, the functionality scanner tells us that the file contains a worm, an IRC backdoor, a downloader, game registry keys, multiple embedded files and an anti-virus process list. According to the database, W32/Spybot is WBspybotCDGMV, which matches every functionality, including the optional ones.

Based on these results, the scanner concludes that this unknown file has strikingly similar statistical data to known W32/Spybot. It would be safe to assume that this file is indeed a W32/Spybot.

If the file is detected by all three scanners, above some preset threshold, then it is assigned the prefix 'certain'. If it is detected by scanner #1, and missed by scanner #2 or #3, then it is assigned the prefix 'suspicious'. If it is detected by #2, and missed by scanner #1 or #3, then it is assigned the prefix 'suspicious'. If it gets detected only by scan #3, then it is assigned the prefix 'guess'.

PERFORMANCE TESTS

The sample set comprised files from *Symantec*'s Sample Management System (SMS) with new signatures added from 1–15 October, all of which are also detected by one other reputable anti-virus scanner. Keep in mind that all the scanning is done with 2,000 lines of heavily commented Perl script, 2,000 lines of mini-signatures, and 35,000 lines of knowledgebase, and it is independent from any existing anti-virus engine or signature database.

Since the threshold level can be customizable for each scanner, my settings for testing were as follows: mini-signature scanning would trigger over a weight sum of 1. If more than one family name was detected, higher certainty rating would arbitrate. Knowledgebase scanning would trigger on over 50% and more than four component matches. Functionality scanner was used as a 'sanity check', and would trigger when all 'necessary' functionality matched.

```
==================================
PWSteal.Bancos, PWSteal.Banpaes, aka PWSteal.Banker
359 files
________________-
302 Certain.Kbancos
 54 Suspicious.Kbancos
  3 Guess.*
  0 No Detect
=================================
W32.Spybot, W32.Gaobot, W32.Randex, W32.Mytob and
other bots, aka Sdbot, Rbot
355 files
________________
295 Certain.*bot
 41 Suspicious.*bot
 12 Guess.*
  7 No Detect
=================================
Clean files
5700 files
________________
  0 Certain.*
  9 Suspicious.*
 85 Guess.*
5606 No Detect
```

AFTERWORDS

Anti-virus scanning at the core is pattern recognition. This algorithm of cooperative scanning using weighted mini-signature and statistical analysis is not restricted to anti-virus scanning alone. Since I am treating each piece of malware like a text file, the same algorithm can be used to categorize any data, especially text documents, given that the categories are well defined. The method could easily be adapted to detect spam, or phishing. Furthermore, with appropriate adaptors, it could categorize all your documents into your personal preferences.

I would like to thank Javier Santoyo for supporting this project, and Douglas Knowles for porting the 'dirty' Perl code to a faster C version.

REFERENCES

[1] William Arnold and Gerald Tesauro, 'Automatically generated Win32 heuristic virus detection', *Proceedings of the 10th Virus Bulletin Conference*, 2000, pp.51–60.

[2] Dr Vesselin Bontchev, 'Current status of the CARO Malware Naming Scheme', *Proceedings of the 15th Virus Bulletin Conference*, 2005, pp.160–170.

FEATURE 3

WHAT IS OUT THERE?

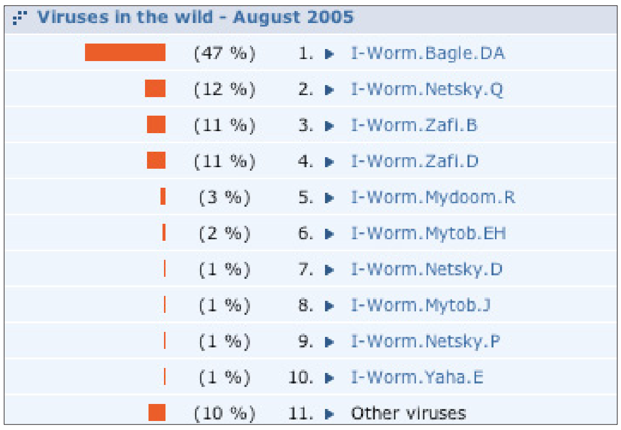
Gabor Szappanos
VirusBuster, Hungary

Do we know what is spreading out there? We, the anti-virus industry, at least pretend that we do. How about our users? From recent experience, it seems that neither we nor they have a truly accurate picture. First, consider the press. Their attention turns only to the latest and most destructive email virus as if it was the greatest threat of them all. Are they to blame? Not really. We are feeding them with data. So, let's turn our attention to ourselves.

The *Virus Bulletin* prevalence table for October 2005 reports the following:

Win32/Mytob	53.21%
Win32/Netsky	36.16%
Win32/Mydoom	5.06%
Win32/Bagle	1.86%
Win32/Zafi	0.74%
Win32/Sdbot	0.69%
Win32/Funlove	0.38%
Others	1.9%

Politically, it might not be the wisest idea to point the finger of blame at *Virus Bulletin* in an article published here. So, are anti-virus vendors any better? One of *VirusBuster*'s recent prevalence lists presented this top ten:



Viruses in the wild - August 2005

(47 %)	1.	I-Worm.Bagle.DA
(12 %)	2.	I-Worm.Netsky.Q
(11 %)	3.	I-Worm.Zafi.B
(11 %)	4.	I-Worm.Zafi.D
(3 %)	5.	I-Worm.Mydoom.R
(2 %)	6.	I-Worm.Mytob.EH
(1 %)	7.	I-Worm.Netsky.D
(1 %)	8.	I-Worm.Mytob.J
(1 %)	9.	I-Worm.Netsky.P
(1 %)	10.	I-Worm.Yaha.E
(10 %)	11.	Other viruses

So, it seems that everyone agrees that email worms rule the known malware universe, with a few Trojan disturbances. In fact, nothing could be further from the truth. Support departments report that the majority of their support calls come from customers experiencing problems caused by Trojans, adware and spyware.

This should be reflected in the statistics, so why isn't it? The problem lies in the methodology we use to collect statistics. The easiest way to do it is to collect information from email gateway logs. But that information relates to *captured* infected emails. One infected PC may send out zillions of infected emails. While the total number of captures is related to the number of infected systems, it is not a one-to-one ratio.

We collect statistics from other sources, as well. This includes user reports and statistics gathered from other types of malware trap. How are these data merged? A user reporting a malware infection represents one infected computer. Meanwhile, 100 captures of Zafi.B may also represent a single infected PC. How do we compare apples with oranges?

Obviously, some form of normalization is needed. The only remaining detail is the question of the normalization factors. Fortunately, in certain cases we have been able to measure it. Apart from spreading in email Zafi.B also performs a DDoS attack on the *VirusBuster* web server (www.virusbuster.hu). This gave us the opportunity to estimate the number of computers infected with Zafi.B.

In a one-hour period on 14 June 2004 we experienced attacks from 105,926 different IP addresses. During the same period *MessageLabs* reported 169,211 infected messages (from 6,459 different hosts). This means that at their measurement point one infected host generated 26.19 messages per day, or 786 infected messages per month.

The raw capture statistics of our email traps for September 2005 are as follows:

I-Worm.Netsky.Q1	523,074
I-Worm.Zafi.B	501,265
I-Worm.Zafi.D	481,479
I-Worm.Mydoom.R	119,185
Trojan.DL.Bagle.DA	90,657
I-Worm.MyTob.H	68,801
I-Worm.Netsky.D3	65,640
I-Worm.Mytob.J	63,074
I-Worm.NetSky.Z	53,729
I-Worm.Yaha.E	45,652
I-Worm.Netsky.P	27,017
I-Worm.Mytob.FV	23,476
I-Worm.Mytob.DR	19,673
I-Worm.Netsky.R	14,991
I-Worm.Mytob.FA	13,060
I-Worm.Netsky.B	12,325
I-Worm.Mytob.FC	11,312
I-Worm.Mytob.DU	10,647
I-Worm.Mytob.IY	10,001
I-Worm.Klez.H	9,773

At this time, our web server logs indicated that there were slightly more than 10,000 Zafi.B infected computers. This means that at this measurement point we have 50 messages per infected host per month. Unfortunately, the normalization factors differ not only with the particular virus variants, but they also vary between different measurement points. Rather hopeless indeed.

Regrettably, we do not have similar data for other email worms. So there is no general solution for obtaining the number of infected PCs from the number of infected messages captured. One could try to extract the original sender from the emails and count all the messages coming from the same PC as only one infection, but that information is not always available to those who gather the statistics.

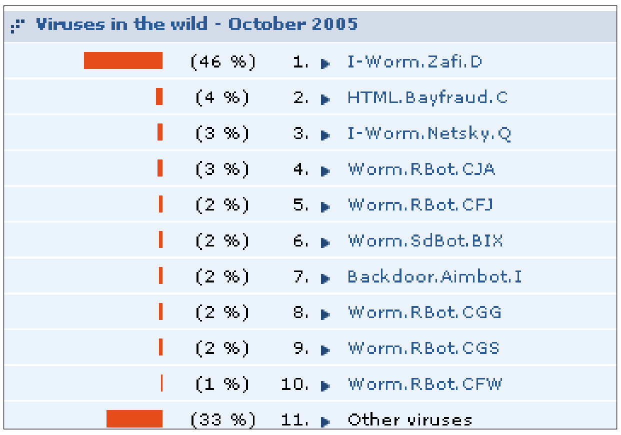
Email is not the only media that is used for spreading viruses. Network worms spread via open (or weakly protected) network shares and/or using *Windows* vulnerabilities. These worms can be captured using SMB traps, or protocol (or rather vulnerability) emulator traps. Protocol emulator traps exist for *Windows* operating systems (*WormRadar*, *iDefense Multipot*, *HBPot*), as well as for x86 *Linux* operating systems (mwcollect, nepenthes). The success of these traps lies in the extent to which they support the shell codes used by different worms, therefore they require periodical updating.

Malware collected with mwcollect in July and August 2005 shows the following distribution:

Worm.RBot.BTW	8.33%
Worm.Rbot.BYE	8.05%
Trojan.DR.Juntador.N	7.26%
Worm.Codbot.Y	5.72%
Worm.RBot.BWY	4.84%
Worm.RBot.BZQ	4.56%
Worm.RBot.BXS	4.51%
Worm.RBot.BWL	3.96%
Worm.RBot.BYD	3.58%
Worm.RBot.BZV	2.93%
Trojan.DR.Juntador.M	2.79%
Worm.RBot.BXR	2.70%
Worm.RBot.CCC	2.23%
Worm.RBot.BZM	2.23%
Trojan.DR.Juntador.D	2.14%
Other	34.16%

It is important to note the appearance of the Juntador droppers, which are self-spreading packages with a custom dropper, usually consisting of an RBot variant (responsible for spreading to other hosts) and an adware installer. Clearly the intention is to use the botnets to install adware packages.

With reasonable normalization of data, and merging the statistics coming from our different traps and user reports, our latest monthly prevalence list looks like this:



Viruses in the wild - October 2005

(46 %)	1.	I-Worm.Zafi.D
(4 %)	2.	HTML.Bayfraud.C
(3 %)	3.	I-Worm.Netsky.Q
(3 %)	4.	Worm.RBot.CJA
(2 %)	5.	Worm.RBot.CFJ
(2 %)	6.	Worm.SdBot.BIX
(2 %)	7.	Backdoor.Aimbot.I
(2 %)	8.	Worm.RBot.CGG
(2 %)	9.	Worm.RBot.CGS
(1 %)	10.	Worm.RBot.CFW
(33 %)	11.	Other viruses

Which is starting to look more realistic.

Is there any hope of getting some real-world data? Yes, we can collect statistics from infected user PCs. There are different approaches for this.

One is to use native worm traps. These are real computers with default OS installations without security patches, connected to the Internet. The great advantage of this approach is that we can collect the samples grabbed by the downloaders as well. The dropped files are also available on these traps. This means that all samples related to an incident are present. The statistics collected from these traps give the best estimate of what the user population is infected with.

Statistics collected in a specific native trap (operated by *Arcabit*) are listed in the table shown on the next page. The table shows that the user population is targeted with a wide range of different malware. The percentage of adware is surprisingly high, indicating that adware is very much an underrated problem.

While this is a very good approach to observing real-world threats, it is still limited in two respects. First, only malware specific to the installed OS version is collected, and secondly, this method does not take into account the active participation of users (such as web browsing, online chat, P2P file exchange, etc.).

It would be useful to carry out sampling on the users' computers. While this is performed by the support department of every AV company, the sample numbers are not high enough to draw any statistically significant conclusions.

At this year's AVAR conference researchers from *Eset* presented statistics collected using their ThreatSense.Net technology. They collect samples that are detected heuristically by their scanner. Obviously, this method is a

2005.08		**2005.09**	
Trojan.Poebot.B	8.73%	Trojan.Poebot.B	16.09%
Trojan.Downloader.Dyfuca.Ei	3.87%	Trojan.Poebot.D	8.91%
Trojan.Lowzones.Hp.S02	3.85%	Trojan.Small.Hp	7.12%
Trojan.Downloader.Agent.Tv	3.74%	Adware.Elitetoolbar.A16	5.28%
Adware.180search.A31	3.26%	Adware.Elitetoolbar.A04.Etb.B2	5.15%
Trojan.Downloader.Istbar.Gen	3.18%	Trojan.Rbot.Gen	4.75%
Trojan.Dubar	3.05%	Trojan.Small.Hp.A16	4.16%
Trojan.Downloader.Agent.Fx	2.89%	Adware.Mediaticket.A16	4.16%
Adware.Mediagtw.A5	2.65%	Trojan.Small.Hp.A01	2.18%
Trojan.Roundstid.Hp	2.55%	Trojan.Hwclk	1.65%
Trojan.Downloader.Small.Asf	2.55%	Adware.Betterinternet.A1	1.65%
Trojan.Nail.B5	2.28%	Trojan.Nanspy.E	1.58%
Adware.Mediagtw.A1	2.20%	Trojan.Rbot.J18	1.45%
Adware.Mediaticket.S05	2.07%	Worm.Gaobot.Gen	1.32%
Trojan.Downloader.Small.Gr	2.04%	Trojan.Rbot	1.19%
Trojan.Poebot.D	1.96%	Adware.Elitetoolbar.A01.A2	1.12%
Trojan.Downloader.Vb.Jl	1.96%	Adware.Elitetoolbar.A01.A1	1.12%
Adware.Elitetoolbar.S02	1.96%	Adware.Clientax.A16	1.12%
Trojan.Rbot.Hp	1.94%	Trojan.Rbot.Hp.A02	1.06%
Adware.Bargainbuddy	1.80%	Adware.Toolbar.Elitebar.Am	1.06%
Other:	41.44%	Other:	27.90%

Statistics collected in a specific native trap (operated by Arcabit).

very biased sampling – the question is can we draw (mostly) unbiased conclusions from it? If we ask the right questions, the answer is yes. According to the latest AV-Comparatives.Org, the *NOD32* scanner is not particularly biased towards Trojans or worms. If we ask only if Trojans or worms are more prevalent on infected computers, then despite the biased sampling the result will be usable.

Win32/TrojanDownloader.Swizzor	43.68%
Win32/Dialer.HZ	7.66%
Win32/Dialer.Q	7.00%
Win32/Dialer	4.61%
Win32/TrojanDownloader.INService	4.54%
Win32/Adware.BetterInternet	3.31%
Worm.RBot	1.65%
Win32/TrojanDownloader.Dluca	1.38%
Win32/Dialer.NAD	1.04%
Win32/Adware.NdotNet	0.87%

The variant names are not of scientific use, but the fact that the majority of the incidents belong to Trojans or adware is another indicator that email worms (or even network worms) are not the biggest threats users face nowadays.

This conclusion is also supported by the long-term tendencies of the different Virus Patrol projects presented at VB2005 by Dmitry Gryaznov: non-replicating malware has taken over from viruses and worms on the Usenet. This does not necessarily indicate the infected state of the user population, but it is a good indication of what is being pushed to them via these additional distribution channels. Clearly, malware distributors are shifting towards Trojans – which can earn them more money than simply playing with self-spreading worms.

CONCLUSION

While it is not possible to find a single source for virus prevalence statistics, useful information can be obtained when statistics are combined from several sources as detailed here. Using all these pieces we still may not be able to put a single prevalence chart on our websites, but we will get a much better understanding of what bothers our users.

All these pieces point to the same conclusion: email worms are no longer the number one threat – non-replicating malware outweigh them in importance. There are positive tendencies though. The WildList has started to include bots, which brings the picture a little closer to reality. Hopefully, vendors will find a way to normalize their statistics to eliminate the over-representation of email worms. Then the press may actually get the real picture and start asking us about the latest terrible destructive adware program – but that will be a different problem.

CONFERENCE REPORT

AVAR 2005: WIRED TO WIRELESS, HACKER TO CYBER-CRIMINAL?

Righard Zwienenberg
Norman, The Netherlands

The 8th annual AVAR conference was held on 17 and 18 November 2005 and took place in Tianjin, China.

During the opening session of the conference, media attention was immense with about a small dozen camera crews shooting footage. Since the morning's programme featured some high ranking state officials, there was also a heavy security presence. In the midst of this crowd, AVAR's chairman Seiji Murakami welcomed all delegates and visitors to the conference, speaking partly in Chinese. It was then over to the conference chair, Mr Zhang Jian, to open the conference.

The first speaker was Xu Jianzhuo from China's Ministry of Public Security, who talked about the current cybercrime situation in China. Like any other country, China has its problems with cybercrime, which is on the increase. Mr Jianzhuo explained that instant messaging is also very popular in China and therefore an easy target, with *QQ* being the most popular messenger. As a nice (if slightly off-topic) anecdote to demonstrate its popularity, Mr Jianzhuo related the story of a farmer who had lost his cow and who used *QQ* to appeal for information, asking any subscriber to contact him if they knew of the cow's whereabouts.

During the rest of the morning's sessions, the audience was briefed on the malware prevalence and Internet security situation in China and South Korea. Chen Mingqi of the national Computer Network Emergency Response Team (CNCERT) gave a presentation focusing on the problems surrounding botnets. An interesting note I took from his presentation was that, under current laws, the use of a command to remove a bot from an affected system could be an illegal action itself if carried out from a remote system. Mr Wankeun Jeon of the Korea Information Security Agency suggested forcing *Windows* to be updated through game sites as one of the new Internet security strategies.

The afternoon started with Ralph Liu presenting a model of how to prevent and manage unknown security threats. He was followed by *Microsoft*'s Jason Garms who gave us an insight into the way in which *Microsoft* assesses ongoing malware prevalence.

Next, Eugene Kaspersky demonstrated that virus writers are now collaborating, both amongst themselves and often with organized crime gangs where they are actively working against anti-virus companies.

On the second day, Vesselin Bontchev informed us about the 'virusability' of *Palm OS*, the risks, the different means of potential infections and the difficulties involved in making an anti-virus product for these devices. Gabor Szappanos explained the pros and cons of worm traps, going into detail about the different types of trap that can be set up. It was interesting to note that in August, Netsky.Q was still the most prevalent virus caught by Gabor's traps.

Candid Wueest confronted attendees with the current threats posed upon online banking. He described the problems with online banking and the situations where the virtual safety is not always actually safe. Even though a connection with a bank might be safe, the information may already have been stolen and sent before it is encrypted. Candid also gave some hints and examples on how to make online banking safer.

François Paget gave an overview of the different ways in which one can become infected with spyware and targeted by adware, what it will do to your system, how it hides itself, and what kind of information is vulnerable. More importantly, he explained several ways in which you can examine your systems to find these critters using freely available utilities. He also demonstrated the basics of removing spyware and adware, but indicated that most of the time this is a complex task.

The conference ended with a panel session in which panellists Eric Ashdown, Dmitry Gryaznov and Guillaume Lovet gave a short presentation about new threats on the Internet, the differences with the past and the way in which the Internet is now exploited by organized criminals using malware, phishing and pharming techniques for monetary gain.

After the closing panel, the traditional conference closing ceremony was held and the venue and organizers of the next AVAR conference were announced: AVAR 2006 will be held in Auckland, New Zealand (dates to be announced in due course at http://www.aavar.org/). The role of conference chairman was passed on to *Eset*'s Randy Abrams for 2006. [*Good luck Randy! - Ed*]

OPINION

THE REAL REASON FOR THE DECLINE OF THE MACRO VIRUS

Dr. Vesselin Bontchev
FRISK Software International, Iceland

In the editorial comment of the December 2005 issue of *Virus Bulletin* (see *VB*, December 2005, p.2), Peter Cooper from *Sophos* makes the correct observation that macro viruses have declined in prevalence since the late 1990s (in fact, the peak was in April 2000). However, his arguments regarding why this has happened are incorrect.

Essentially, he attributes this decline to the following factors:

- Users have become more 'security-aware' (and are, apparently, no longer opening *Office* documents from unknown sources).
- Virus writers have become frustrated with the limitations of the macro language.
- Email servers are blocking *Office* attachments from unknown senders.

Let me debunk briefly each of the above arguments and then I shall present the *real* reason why macro malware has declined in prevalence.

THE BOGUS REASONS

The claim that users have somehow become more 'security-aware' is preposterous. Binary email worms that infect the victim's machine only after the user double-clicks on the executable email attachment containing the virus are still quite prevalent. Have the users somehow learned not to double-click on attached *Office* documents, while being unable to stop double-clicking on executable attachments? Hardly.

'Face it, folks, if user education was going to work, IT WOULD HAVE WORKED BY NOW!*'*

Users are just as stupid and ignorant as they were five years ago. In fact, I have a message for anybody who still harbours any illusions that user education can work to solve the virus problem on a large scale: face it, folks, if user education was going to work, *IT WOULD HAVE WORKED BY NOW*!

The claim that the macro language is 'too limiting' could be made only by someone who has insufficient experience in this field. Visual Basic for Applications (VBA) is an extremely powerful language which provides everything necessary to write a sophisticated and successful virus.

Anybody who has any interest in the computer virus field will certainly remember how widespread the Melissa virus became – and since then there have been a number of far more sophisticated (although less well publicized) viruses that have used complex polymorphic and spreading techniques.

While there *are* a few tasks that VBA is unable to perform directly (e.g. achieve memory residency, use rootkit techniques, etc.), absolutely nothing prevents VBA malware from dropping 32-bit *Windows* executable components that could do the rest – and, indeed, many macro viruses do use this approach.

While I don't have much experience designing email servers, I have yet to encounter a single one that filters out *Office* attachments 'from unknown senders'. Users routinely send *Word* documents and *Excel* spreadsheets to each other as email attachments and would scream if these attachments were to be gobbled by the email server. And how would the email server know whether the sender is unknown to the recipient, anyway?

The *real* reason why macro malware has declined in prevalence is very different.

THE REAL REASON

In *Office 2000*, *Microsoft* introduced a security measure, based on an idea suggested several years ago by a number of anti-virus experts, including the author of these lines. While this security measure was not turned on by default in *Office 2000*, in the later versions of *Office* it was – which finally made it effective and caused, if not the complete disappearance, at least the significant reduction of the macro malware threat.

The security measure consisted of the following. When it was turned on, the *Office* application would *silently* ignore any macros that were not digitally signed with a key, marked as trusted by the user.

The emphasis on the word 'silently' is intentional and it was crucial for the success of this security measure. If, instead, the *Office* applications had just displayed a warning, allowing the user to choose whether to run or ignore the macros present in the document, most users would have clicked mindlessly on the 'OK' button anyway. But with the macros ignored silently, users weren't (and aren't) even aware that they are present, so they weren't (and aren't) tempted to let them run.

This measure relies on the fact that while users often exchange *Office* documents, they very rarely exchange macros. Many users *use* macros – but either they use macros that they have written themselves, or they use macro packages they have purchased from legitimate vendors.

Microsoft provided a tool for digitally signing macros, which meant that the macro package developers could sign their packages before distribution and users could easily sign any macros they have written themselves. Of course, virus writers were free to do the same, but unlike the macros produced by legitimate developers or the ones written by the user, the macros created by the virus writers wouldn't be signed with a key marked as trusted by the user. So, while a few macro virus writers did try to distribute digitally-signed viruses, they were unable to achieve any significant success.

SUFFICIENTLY EFFECTIVE

Microsoft has introduced many other security measures in *Office* too – such as forbidding a VBA project from copying itself from the global template to documents via 'standard' means; preventing a VBA project from accessing itself in some *Office* applications; introducing a warning if a document contains macros; making it difficult to send multiple emails via *Outlook* from a macro; and so on.

'The "by default, only macros signed with a trusted key are allowed to run, all others are silently ignored" policy turned out to be the key solution.'

However, all of these measures have turned out either to be easy for virus writers to circumvent or fail otherwise to protect the user. Unlike them, the 'by default, only macros signed with a trusted key are allowed to run, all others are silently ignored' policy turned out to be the key solution that essentially turned macro malware from a major threat into a minor annoyance.

While this policy is not infallible (for instance, the setting can easily be turned off by changing a registry key – but any program that can do that must be allowed to run first, so cannot be a macro), it turned out to be sufficiently effective against macro viruses.

It is a great pity that the same policy cannot be applied successfully to executable files – because, sadly, users still download executables from dubious sources and still send executables (e.g., screen savers) to each other.

CALL FOR PAPERS

VB2006 MONTRÉAL

Virus Bulletin is seeking submissions from those wishing to present at VB2006, the Sixteenth Virus Bulletin International Conference, which will take place 11–13 October 2006 at the Fairmont The Queen Elizabeth, Montréal, Canada.

The conference will include three days of 40-minute presentations running in two concurrent streams: Technical and Corporate.

SUGGESTED TOPICS

Submissions are invited on all subjects relevant to anti-malware and anti-spam. In particular, *VB* welcomes the submission of papers that will provide delegates with ideas, advice and/or practical techniques, and encourages presentations that include practical demonstrations of techniques or new technologies.

A list of topics suggested by the attendees of VB2005 can be found at http://www.virusbtn.com/conference/vb2006/2006call.xml. However, please note that this list is not exhaustive, and the selection committee will consider papers on these and any other anti-malware and spam-related subjects.

HOW TO SUBMIT A PAPER

Abstracts of approximately 200 words must be sent as plain text files to editor@virusbtn.com, to arrive no later than **Thursday 9 March 2006**. Submissions received after this date will not be considered. Please include full contact details with each submission.

Following the close of the call for papers all submissions will be anonymised before being reviewed by the selection committee; authors will be notified of the status of their paper by email.

Those whose submissions are accepted for the conference programme are required to provide a paper that will be published in the conference proceedings. Authors are advised in advance that the deadline for submission of the completed papers will be Monday 5 June 2006, and that full papers should not exceed 6,000 words.

Further details of the paper submission and selection process are available at http://www.virusbtn.com/conference/.

PRODUCT REVIEW

WINDOWS ONECARE LIVE BETA (BUILD 0.8.0794.44)

Matt Ham

It has been something of an open secret for a while that some form of *Microsoft* anti-virus is on its way, though further details have been less forthcoming. Even now, after the beta release of *Windows OneCare Live*, the area of *Microsoft*'s main website dedicated to security has no obvious links to the product – which is somewhat mystifying since reference to the company's anti-spyware product (also in beta) is hard to miss.

As many readers will remember, there was a previous *Microsoft* anti-virus product, which was released more than ten years ago. However, the product in question was doomed to vilification since it remained unchanged during a long period of OS production and there was no easy method of updating it. However, times have moved on and it is to be expected that the new beta will make full use of today's prevalence of Internet connections for retrieving its updates.

For *Microsoft*, OS integration is not the simple matter it is for many producers, since legal as well as functional issues come into play. Complete separation of the product from the OS would be likely to put off potential users, while too tight a binding to the internal workings and external tools could see *Microsoft* charged with unfair competitive practices.

Thus before settling down with the product I already had a few expectations and uncertainties. Since the release of *Windows XP Service Pack 2*, the *Windows Security Center* (*WSC*) has been the hub of security configuration and status reporting on the *XP* platform. I would expect any anti-virus functionality to integrate into the *WSC* at the very least, but total replacement of the *WSC*, as in *McAfee*'s consumer product, seemed unlikely (thus avoiding accusations of elbowing out competing products that do use the *WSC*).

Perhaps of more concern was the question of how the anti-virus and anti-spyware products would co-exist. The anti-spyware beta application is fully standalone. However, both products contain on-access scanners and both might conceivably attempt to detect files that are related to spyware and demonstrate viral replication. The anti-spyware functionality includes some fairly extensive behaviour blocking in addition, some of which may be anti-viral in effect, if not in intention.

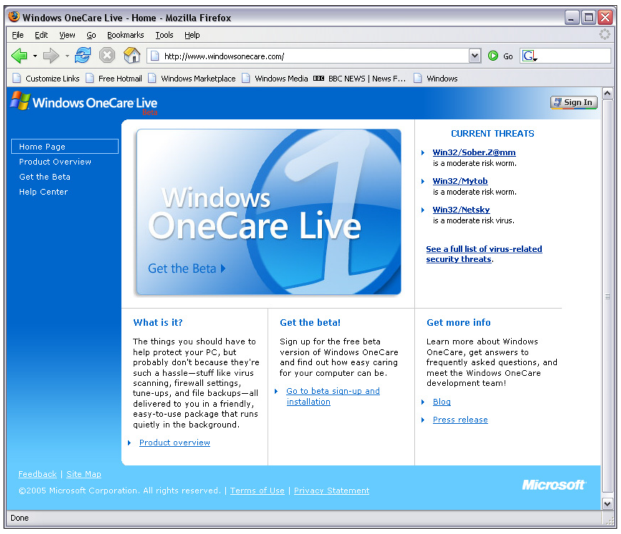
With these thoughts in mind I proceeded to the somewhat hidden *OneCare* beta site. I must stress at this stage that the product *is* in beta and thus major changes may occur even between the time I write this review and the time it is published. Comments here are thus an initial view and are by no means (even less than usual) intended to be a buyers' guide to the product.

The version I tested was updated several times during the testing process, with both the base product and the definitions being changed. All comments made here refer to the final version that was installed. Newer versions may, of course, differ considerably. The operating system used was *Windows XP Professional Service Pack 2*, with all updates applied using *Windows Update*. *Windows OneCare* is available from http://beta.windowsonecare.com/.

INSTALLATION AND UPDATE

The use of *Firefox* on web-based installation routines seems to be a method destined to failure. This fussiness does not come as a surprise at beta stage, but I fear the situation will change little in the full release if my recent experiences with other developers are typical. In this case the installation platform is not checked until after the user has supplied an email address – so it seems that those who refuse to use *Internet Explorer* will be subjected to emails concerning the beta despite not being able to install the software.

Thus I reverted to my copy of *Internet Explorer*, which in the past has been subject to a variety of extremely lax security settings, all of which were demanded for the installation of security products. However, in this case there is no need to change settings from the application's default values, which is a relief on the security front. It is noted before installation begins that automatic updates are activated as a part of the installation process, not only for *OneCare* but for all *Microsoft* products. This is not a bad default setting, and I am surprised it was not made into a difficult-to-disable default within *SP2*.



CURRENT THREATS

- Win32/Sober.Z@mm is a moderate risk worm.
- Win32/Mytob is a moderate risk worm.
- Win32/Netsky is a moderate risk virus.

See a full list of virus-related security threats.

What is it?

The things you should have to help protect your PC, but probably don't because they're such a hassle—stuff like virus scanning, firewall settings, tune-ups, and file backups—all delivered to you in a friendly, easy-to-use package that runs quietly in the background.

- Product overview

Get the beta!

Sign up for the free beta version of Windows OneCare and find out how easy caring for your computer can be.

- Go to beta sign-up and installation

Get more info

Learn more about Windows OneCare, get answers to frequently asked questions, and meet the Windows OneCare development team!

- Blog
- Press release

Feedback | Site Map

©2005 Microsoft Corporation. All rights reserved. | Terms of Use | Privacy Statement

Microsoft

The next stage is an automatic check that the machine in question reaches installation requirements – though no information is given as to what is being checked. However, any problems relating to browsers and *OneCare* already being installed are picked up at this point. At the moment *OneCare* does not support updating by overwriting an existing *OneCare* installation, though this is likely to change with the full release. If it doesn't, there is the potential for chaos should uninstallation files for *OneCare* be corrupted and *OneCare* itself also be in need of reinstallation. Perhaps not very likely through fate alone, but not entirely unlikely if malware is targeting the application.

After this check comes another, this time requiring an ActiveX application in order to check machine hardware and software details. Privacy fanatics will note that this gathers and sends various pieces of personal information as it executes, with the usual 'we will not share this information' caveat being given in the small print. This is also the stage at which any other anti-virus products on the system are detected and declared to be bad things which must be expunged. Once the ActiveX has been approved and installed the process here is fast and leads, if the checks are all passed, to the licence agreement.

After this, files are downloaded for installation. Installation is confirmed by the presence of a small green icon on the task bar. Many of the status notifications are also seen here, in the form of the infamous popup information bubbles.

The add/remove programs dialog gives the size of the *OneCare* package as 129 MB, though the initial download is certainly smaller than this with typical download times of only a few minutes at 1 MB/s download speeds. After download, the installation process is almost instant and requires only a reboot for completion. No other settings are tweakable here, which, again, may be something that is unique to the beta version. One feature of the method of installation used here is that installation onto standalone machines is currently not possible. Installation requires an active Internet connection.

Uninstallation of *OneCare* is short but simple and understandably requires a reboot as part of the process. The reboot can be delayed, which may be convenient, but this does result in a machine with no active firewall or anti-virus software until the reboot is performed. The reboot dialog states that network connections may fail until a reboot has been performed, though this did not seem to be the case. It would be better, from a security point of view, if network connections were terminated until reboot.

As might be expected, updates are automatic, although the process is rather mysterious. In this version there are no options beyond what amount to: updates on, updates off and look for updates now. These three are certainly enough, though the lack of an ability to set parameters for updates seems very strange when one is accustomed to dealing with the ultra-tweakable anti-virus packages of today.

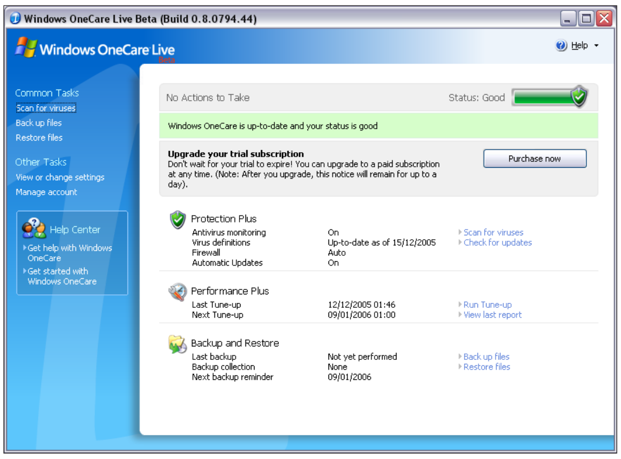
Updates are not limited to those which affect *OneCare* directly. Urgent *Windows* updates are also checked, effectively duplicating some of the functionality of *Windows Update*.

FEATURES

The mention of the update mechanisms just made will give some idea as to how much the product interacts with other established parts of the operating system. The main interface, for example, bears more than a passing resemblance to the *Windows Security Center* due to the presence of a large green (or red) shield in a three-part list of functions. In fact, the resemblance between the two interfaces is not so great upon closer inspection but is enhanced by the left-hand control pane, right-hand status view configuration.

The left-hand pane contains various actions and is, in effect, a set of short cuts to various likely actions and links to useful sources of information. Documentation is not yet in the form of a context-sensitive help system but rather through hyperlinks within the various dialogs. The links here take the user directly to the highest level entries in the help resources. The View or Change Settings Dialog is also positioned here, though to reduce confusion, the configuration changes possible here are included when the functionality is discussed in the right-pane description.

The right-hand pane is dominated by the status slider, which in my experience had only two positions – green where everything is approved in its settings, and red when any security setting is lessened. There is also an option to upgrade the subscription under which *OneCare* is being

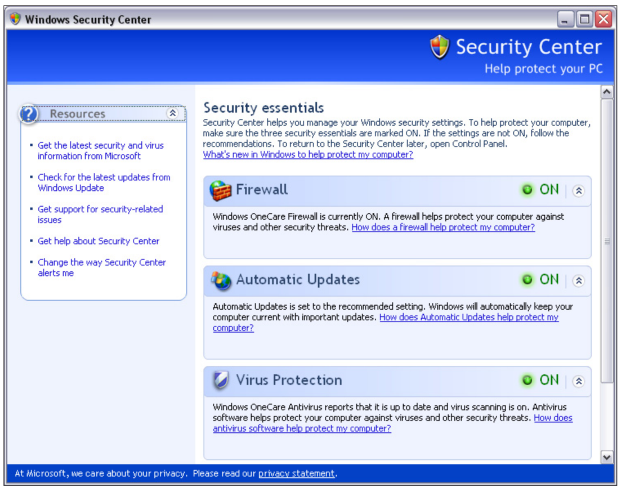


No Actions to Take Status: Good

Windows OneCare is up-to-date and your status is good

Upgrade your trial subscription
Don't wait for your trial to expire! You can upgrade to a paid subscription at any time. (Note: After you upgrade, this notice will remain for up to a day).

Protection Plus	
Antivirus monitoring	On
Virus definitions	Up-to-date as of 15/12/2005
Firewall	Auto
Automatic Updates	On
Performance Plus	
Last Tune-up	12/12/2005 01:46
Next Tune-up	09/01/2006 01:00
Backup and Restore	
Last backup	Not yet performed
Backup collection	None
Next backup reminder	09/01/2006


Windows Security Center

Security Center
Help protect your PC

Resources

- Get the latest security and virus information from Microsoft
- Check for the latest updates from Windows Update
- Get support for security-related issues
- Get help about Security Center
- Change the way Security Center alerts me

Security essentials

Security Center helps you manage your Windows security settings. To help protect your computer, make sure the three security essentials are marked ON. If the settings are not ON, follow the recommendations. To return to the Security Center later, open Control Panel.
What's new in Windows to help protect my computer?

Firewall ON

Windows OneCare Firewall is currently ON. A firewall helps protect your computer against viruses and other security threats. How does a firewall help protect my computer?

Automatic Updates ON

Automatic Updates is set to the recommended setting. Windows will automatically keep your computer current with important updates. How does Automatic Updates help protect my computer?

Virus Protection ON

Windows OneCare Antivirus reports that it is up to date and virus scanning is on. Antivirus software helps protect your computer against viruses and other security threats. How does antivirus software help protect my computer?

At Microsoft, we care about your privacy. Please read our privacy statement.

used, (obviously not relevant in the beta product). The three areas below this for which status is shown are Protection Plus, Performance Plus and Backup and Restore. For the purposes of this overview the latter two were not given much attention, though it is appropriate to mention them before heading to the more overtly security-relevant portions.

Performance Plus contains one anti-virus related function: that of a scheduled scan of the machine. However, this is combined with yet another scheduled check for updates, defragmentation of disks if required, automatic run of the backup application and an optional cleanup – deleting files that are presumably unnecessary. Each of these processes on its own can be something of a system hog. Combining these is clearly going to make the affair very burdensome indeed on system resources and the documentation for this area states that it might be best to allow the process to run unattended. Having seen the effect of such a tune-up in progress, I can say that this ‘recommendation’ is actually a necessity on all but very high-end machines.

Backup and Restore does very much what its name suggests. Backups are very useful in cases where files have been devastated by malware, though this particular backup method does seem somewhat limited in that it requires *OneCare* to be present for the retrieval of files. This makes the functionality less useful for system recovery and rather more for pure data recovery.

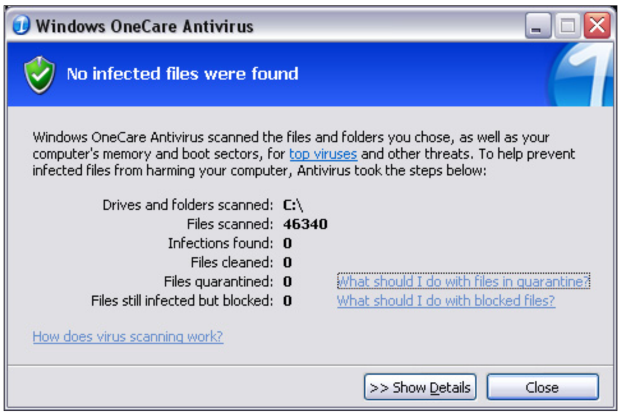
Protection Plus consists of on-access scanning, virus definition updates and, yet again, general updates for *OneCare*. It certainly seems that updates are one of the most stressed aspects of the application. Given the number of non-updated anti-virus applications that have given false security over the years, this is definitely a good thing (although, ironically, one of the first sources of this false security was the original *Microsoft AntiVirus*). Somewhat surprisingly this is an area where there is very little to comment upon and even less to see. On-access scanning is either on or off as a general setting. In addition, heuristics may be disabled, the default state being enabled.

It is also possible to specify exceptions for scanning here, though it seems that whole drives cannot be selected, nor indeed can anything but individual files. This is a simplistic and potentially irritating way of dealing with exclusions, though likely to encourage exclusions for only a very small number of files.

It is obvious from this list of configurable features that the anti-virus functionality is not really designed to be messed around with, the user having to be content that *Microsoft* has chosen appropriate settings. In all honesty this is probably the best way to deal with the majority of customers. Unfortunately a vocal minority is almost certain to complain about the lack of features, believing that more is always better, even for the most technically philistine user.

The firewall function of *OneCare* replaces rather than complementing the existing *Windows Firewall*, which surprised me. I certainly can’t see people being persuaded to buy *OneCare* based on it having a firewall included when one is available already, free of charge, from the same vendor. This aside, the *OneCare* firewall integrates with the WSC as would be expected. Firewall settings are controlled through the same menu as the anti-virus functionality already mentioned.

Three sensitivity settings are provided. The first one, off, requires little explanation. The automatic setting is the default, which allows programs access to the Internet if they are known or are given explicit permission. Permission may be granted manually through this interface or, more usually, when the application is first used. Permissions may be enabled or disabled on an individual basis through the GUI, or deleted totally. Blocking here occurs immediately if permissions are rescinded, even if the connection has


Windows OneCare Antivirus

No infected files were found

Windows OneCare Antivirus scanned the files and folders you chose, as well as your computer's memory and boot sectors, for top viruses and other threats. To help prevent infected files from harming your computer, Antivirus took the steps below:

```
        Drives and folders scanned:  C:\
                     Files scanned:  46340
                  Infections found:  0
                     Files cleaned:  0
                 Files quarantined:  0     What should I do with files in quarantine?
   Files still infected but blocked:  0     What should I do with blocked files?
```

How does virus scanning work?

already been opened by the application in question. The more stringent Prompt setting does not allow any program to access the net without manual permission on every occasion.

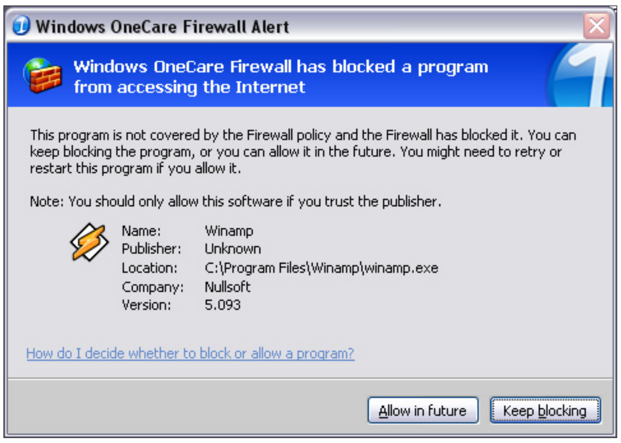
The firewall settings are slightly different from those supplied by the built-in *Windows Firewall*, in both good and irritating ways. For example, exceptions from blocking are not simply triggered by filename but are rechecked if an application is patched or overwritten. This seals a major flaw in the old firewall, where malware could simply replace an allowed application and trigger no alerts when accessing the net.

On the more irritating side is the fact that *Microsoft* products – *Windows AntiSpyware* and *Internet Explorer* for example – are allowed access to the net automatically, while their non-*Microsoft* equivalents (*Firefox* being a prime example) cause warnings to erupt on first use. Obviously *Microsoft* is able to verify the trustworthiness of its own applications, with other products being less easily declared as benign. However, it would be nice at some point for *OneCare* to include a list of commonly used exceptions for blocking which were not necessarily created by *Microsoft*.

More interestingly, although *OneCare* is aware of *Windows AntiSpyware* the reverse is not the case. During initial use of *OneCare* several of its activities were flagged as worthy of attention by *AntiSpyware*'s behaviour analysis.

SCANNING

When run against the *Virus Bulletin* In the Wild test set as used in the most recent comparative review (see *VB*, December 2005, p. 16), results were initially difficult to determine due to the lack of any real logging. However, after further investigation it was determined that the scanner had missed detection of one sample of W32/Argobot and two samples of W32/Mytob.


Windows OneCare Firewall Alert

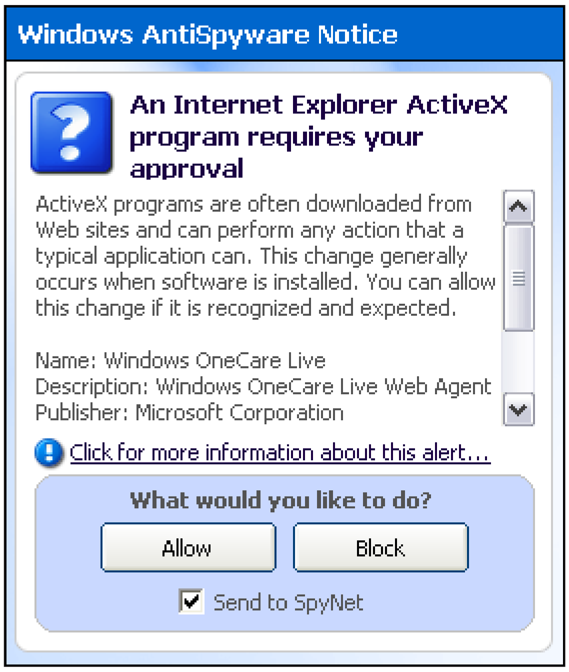
Windows OneCare Firewall has blocked a program from accessing the Internet

This program is not covered by the Firewall policy and the Firewall has blocked it. You can keep blocking the program, or you can allow it in the future. You might need to retry or restart this program if you allow it.

Note: You should only allow this software if you trust the publisher.

Name: Winamp
Publisher: Unknown
Location: C:\Program Files\Winamp\winamp.exe
Company: Nullsoft
Version: 5.093

How do I decide whether to block or allow a program?


Windows AntiSpyware Notice

An Internet Explorer ActiveX program requires your approval

ActiveX programs are often downloaded from Web sites and can perform any action that a typical application can. This change generally occurs when software is installed. You can allow this change if it is recognized and expected.

Name: Windows OneCare Live
Description: Windows OneCare Live Web Agent
Publisher: Microsoft Corporation

Click for more information about this alert...

What would you like to do?

CONCLUSION

Conclusions seem a little premature, though some general thoughts occurred to me while inspecting the features of *Windows OneCare*. First is that this is obviously designed to be as simple as possible to operate. The installation procedure is very much click-and-forget and the update procedures are not only automated but improve upon the updates for other *Windows* components. Certainly it makes sense for the anti-virus application to ensure that patches are applied before a virus is on the machine rather than taking action when the machine is infected.

Questions remain, however, as to how the product will interact with existing security applications. Including anti-virus and firewall functions in *OneCare*, while having a completely separate anti-spyware product seems unneccesarily confusing. The similarity between the *OneCare* interface and the *Windows Security Center* might mean that it will be either merged with, or replace, that application, or the similarity might simply be an effort to maintain a coherent look and feel. Thankfully, the beta state of the product makes answers to these questions impossible, so I can sleep soundly despite not knowing the answers.

Technical Details

Test environment: AMD64 3800+ machine with 1 GB RAM, 80 GB hard disk, DVD/CD-ROM and 1 MBit ADSL Internet connection running *Windows XP Professional SP2*.

Developer: *Microsoft*, One Microsoft Way, Redmond, WA 98052, USA. Website http://beta.windowsonecare.com/.

END NOTES & NEWS

The inaugural AVIEN/AVIEWS conference will take place from 11am to 4pm Eastern Standard Time on 18 January 2006 by webcast. Registration for the conference is available at http://www2.nortel.com/go/events_detail.jsp?cat_id=-8004&oid=100191141.

The Black Hat Federal Briefings & Training takes place 23–25 January 2006 in Washington, DC, USA. Registration for the event is now open. See http://www.blackhat.com/.

The Second Annual IFIP WG 11.9 International Conference on Digital Forensics will take place 29 January to 1 February 2006 in Orlando, FL, USA. Technical papers on all areas related to the theory and practice of digital forensics will be presented. For more details see http://www.cis.utulsa.edu/ifip119/.

IT-DEFENSE 2006 takes place 30 January to 3 February 2006 in Dresden, Germany. The event is aimed at network administrators, developers, DP managers, IT security officers, auditors, consultants and hackers seeking to exchange information and make contact with leaders in the industry. For more information see http://www.it-defense.de/.

Techworld is organising a free, half-day seminar on 2 February 2006 in London, UK. The seminar, entitled 'Endpoint Security: how to protect your system from its users', will focus on areas such as threat trends, patch management, securing endpoints, virus protection and configuration management. See http://www.techworld.com/.

RSA Conference 2006 will be held 13–17 February 2006 in San Jose, CA, USA. For more details including the full agenda and online registration see http://2006.rsaconference.com/us/.

The Black Hat Europe 2006 Briefings & Training will be held 28 February to 3 March 2006 in Amsterdam, The Netherlands. For details including online registration see http://www.blackhat.com/.

The 9th annual WEBSEC conference takes place 27–31 March 2006 in London, UK. The event will include live hacking demos, a network and application hacker challenge, more than 40 sessions on topical security issues including a panel debate in which *Virus Bulletin*'s Technical Consultant Matthew Ham will participate. For more details see http://www.mistieurope.com/.

Infosecurity Europe 2006 takes place 25–27 April 2006 in London, UK. For details or to register interest in the event see http://www.infosec.co.uk/.

The 15th EICAR conference will take place from 29 April to 2 May 2006 in Hamburg, Germany. Authors are invited to submit full papers and posters for the conference. The deadlines for submissions are as follows: academic papers (in full) 13 January 2006; poster presentations 24 February 2006. For more information see http://conference.eicar.org/2006/.

The Seventh National Information Security Conference (NISC 7) will take place from 17–19 May 2006 at St. Andrews Bay Golf Resort & Spa, Scotland. Enquiries may be directed to tina.deighton@sapphire.net or via http://www.nisc.org.uk/ .

The Fourth International Workshop on Security in Information Systems, WOSIS-2006, will be held 23–24 May 2006 in Paphos, Cyprus. For details see http://www.iceis.org/.

CSI NetSec '06 takes place 12–14 June 2006 in Scottsdale, AZ, USA. Topics to be covered at the event include: wireless, remote access, attacks and countermeasures, intrusion prevention, forensics and current trends. For more details see http://www.gocsi.com/.

Black Hat USA 2006 will be held 29 July to 3 August 2006 in Las Vegas, NV, USA. A call for papers will open on 2 February and online registration for the event will be available from 15 March. See http://www.blackhat.com/.

The 16th Virus Bulletin International Conference, VB2006, will take place 11–13 October 2006 in Montréal, Québec, Canada. *Virus Bulletin* is currently seeking submissions from those interested in presenting papers at the conference – see p.15 for the full call for papers. For details of sponsorship opportunities, please email vb2006@virusbtn.com. Online registration and further details will be available soon at http://www.virusbtn.com/.

SUBSCRIPTION RATES

Subscription price for 1 year (12 issues):

- Single user: $175
- Corporate (turnover < $10 million): $500
- Corporate (turnover < $100 million): $1,000
- Corporate (turnover > $100 million): $2,000

See http://www.virusbtn.com/virusbulletin/subscriptions/ for subscription terms and conditions.

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 Web: http://www.virusbtn.com/

vb Spam supplement

CONTENTS

NEWS & EVENTS

NEW ANTI-SPAM GROUP FOR CHINA

The Internet Society of China (ISC) has announced the formation of a new group focusing on the problem of spam in China – the Anti-Spam Committee of the Internet Society of China (ASISC). The new group will work alongside the existing ISC Anti-Spam Coordination Team, although the ASISC is expected to be more industry-driven than the current team.

The main responsibilities of the ASISC will be to institute email service standards and criteria, protect email users' legitimate rights, develop international cooperation and improve email service quality. ASISC currently consists of China's largest email and network service providers, relevant corporations, enterprises, scientific research institutes and government sectors that are concerned with promoting the development of the email service industry.

ATTEMPT TO BAN VOICEMAIL SPAMMING

Canadian wireless operator *Rogers Wireless Inc.* is appealing to the Canadian Radio-television and Telecommunications Commission (CRTC) to ban the marketing practice of voicemail spamming.

The process the company objects to is known as 'voicecasting' and involves the use of an automated dialling device to transmit marketing messages directly into the voicemail accounts of mobile telephones. Because the messages are transmitted directly to the voicemail accounts, the telephone does not ring, so consumers don't realise they've received the advertising messages until they retrieve their voicemail.

In a complaint filed with the CRTC, *Rogers Wireless* argued that mobile phone customers should not have to pay airtime charges to access marketing messages that have been disguised as legitimate voicemail. The company also pointed out that the costs are even higher for customers who retrieve messages while abroad, when roaming or long-distance charges also come into play.

Somewhat surprisingly, the man credited with inventing voicemail spamming says he agrees with *Rogers Wireless*. Cesar Correia, founder of *Infolink Technologies Inc.*, the largest provider of voicecasting services in Canada, said: 'Customers should not be incurring charges when somebody is sending a voicecast to a cellphone. It's unfair to the public.'

Telephone companies and consumer groups will be hoping that *Rogers Wireless* has put together a strong case – in 2001 *Bell* made a similar complaint to the CRTC about voicecasting, but the regulator ruled that there was insufficient evidence to suggest that the practice was a nuisance to consumers. This time the CRTC is expected to issue a public notice and seek comment on the issue.

SPAMMERS SUED

A company found guilty of spamming has been ordered to pay $3 million in civil penalties and $375,000 in restitution to Seattle Public Schools. The ruling against Californian marketing firm *AvTech Direct* resulted from Washington state's first lawsuit under the federal anti-spam act. According to the state, *AvTech Direct* sent unsolicited emails advertising the sale of desktop computers to thousands of consumers, including 1,500 to school district employees. In addition to the penalties, *AvTech Direct*, is prohibited from engaging in similar practices in Washington.

Meanwhile, a man has been sued by the state of North Carolina in its first anti-spam case. Michael Abbott is accused of sending hundreds of unsolicited emails touting a phoney fuel booster. The complaint alleges that Abbot used bogus return addresses and failed to provide a reliable opt-out mechanism. He faces fines up to $5,000 for each violation. According to a state Justice Department spokesperson, North Carolina officials were given a tip-off about Abbott's practices by the Federal Trade Commission (FTC), which in turn had been given information by *Microsoft* after the company captured some of Abbott's messages in its spam traps.

FTC SAYS CAN-SPAM CAN

The US Federal Trade Commission (FTC) has reported to Congress that the two-year-old CAN-SPAM Act is effective in providing protection for consumers, and that it is being enforced aggressively by state and federal law enforcers as well as the private sector.

Despite concluding that the Act is effective, the FTC suggests three steps that would improve its efficacy still further. First, the report recommends that Congress should enact the 'US SAFE WEB Act', to improve the FTC's ability to trace spammers who operate outside of the US. Second, it recommends the continuation of efforts to educate consumers about spam. Finally, the report calls for continued work in the development and advancement of anti-spam technology, and in particular, tools that prevent spammers from operating anonymously.

EVENTS

The 6th general meeting of the Messaging Anti-Abuse Working Group (MAAWG) will take place 28 February to 2 March 2006 in San Francisco, CA, USA. Members and non-members are welcome. For details see http://www.maawg.org/.

The Authentication Summit II takes place on 19 April 2006 in Chicago, IL, USA. The conference will cover the latest advances in email authentication, including Sender ID Framework (SIDF) and DomainKeys Identified Mail (DKIM), with a focus on real-life results and prescriptive information. For full details see http://emailauthentication.org/.

INBOX 2006 will be held 31 May to 1 June 2006 in San Jose, CA, USA. The event will cover all aspects of email including topics such as 'has CAN-SPAM failed us?', 'what can ISPs do to fix spam?', 'how not to be a spammer' and 'new directions in identifying spam'. For more information see http://www.inboxevent.com/2006/.

The third Conference on Email and Anti-Spam, CEAS 2006, will be held 27–28 July 2006 in Mountain View, CA, USA. The conference encompasses a broad range of issues relating to email and Internet communication. Those wishing to present papers are invited to submit their proposals before 23 March 2006. For full details see http://www.ceas.cc/.

The Text Retrieval Conference (TREC) 2006 will be held 14–17 November 2006 at NIST in Gaithersburg, MD, USA. As in 2005, TREC 2006 will include a spam track, the goal of which is to provide a standard evaluation of current and proposed spam filtering approaches, thereby laying the foundation for the evaluation of more general email filtering and retrieval tasks. For more information see http://plg.uwaterloo.ca/~gvcormac/spam/.

FEATURE

THE TREC 2005 SPAM FILTER EVALUATION TRACK

Gordon V. Cormack
University of Waterloo, Canada

The 14th Text Retrieval Conference (TREC 2005) took place in November 2005. One of the highlights of the event was the TREC spam filter evaluation effort, in which 53 spam filters developed by 17 organizations were tested on four separate email corpora totalling 318,482 messages. The purpose of the exercise was not to identify the best entry; rather to provide a laboratory setting for controlled experiments. The results provide a helpful insight into how well spam filters work, and which techniques are worthy of further investigation.

The technique of using data compression models to classify messages was demonstrated to be particularly worthy, as evidenced by the fine performance of filters submitted by Andrej Bratko and Bogdan Filipi of the Josef Stefan Institute in Slovenia. Other techniques of note are the toolkit approach of Bill Yerazunis' *CRM114* group and the combination of weak and strong filters by Richard Segal of *IBM*.

EVALUATION TOOLS & CORPORA

Each spam filter was run in a controlled environment simulating personal spam filter use. A sequence of messages was presented to the filter, one message at a time, using a standard command-line interface. The filter was required to return two results for each message: a binary classification (spam or ham [not spam]), and a 'spamminess' score (a real number representing the estimated likelihood that the message is spam). After returning this pair of results, the filter was presented with the correct classification, thus simulating ideal user feedback. The software created for this purpose – the TREC Spam Filter Evaluation Tool Kit – is available for download under the Gnu General Public License (http://plg.uwaterloo.ca/~trlynam/spamjig/).

The spam track departed from TREC tradition by testing with both public and private corpora. The public corpus tests were carried out by participants, while the private corpus tests were carried out by the corpus proprietors. This hybrid approach required participants both to run their filters, and to submit their implementations for evaluation by third parties.

The departure from tradition was occasioned by privacy issues which make it difficult to create a realistic email corpus. It is a simple matter to capture all the email delivered to a recipient or set of recipients. However, acquiring their permission, and that of their correspondents, to publish the email is nearly impossible. This leaves us with a choice between using an artificial public collection and using a more realistic private collection. The use of both strategies allowed us to investigate this trade off.

The three private corpora each consisted of all the email (both ham and spam) received by an individual over a specific period. The public corpus came from two sources: *Enron* email released during the course of the Federal Energy Regulatory Commission's investigation, and recent spam from a public archive. The spam was altered carefully so as to appear to have been addressed to the same recipients and delivered to the same mail servers during the same period as the *Enron* email. Despite some trepidation that evidence of this forgery would be detected by one or more of the filters, the results indicate that this did not happen.

To form a test corpus, each message must be augmented with a gold standard representing the true classification of a message as ham or spam. The gold standard is used in simulating user feedback and in evaluating the filter's effectiveness. As reported previously (see *VB*, May 2005, p.S1), much effort was put into ensuring that the gold standard was sufficiently accurate and unbiased. Lack of user complaint is insufficient evidence that a message has been classified correctly. Similarly, we believe that deleting hard-to-classify messages from the corpus introduces unacceptable bias. In comparing the TREC results with others, one must consider that these and other evaluation errors may tend to overestimate filter performance.

EVALUATION MEASURES

The primary measures of classification performance are ham misclassification percentage (hm%) and spam misclassification percentage (sm%). A filter makes a trade-off between its performances on these two measures. It is an easy matter to reduce hm% at the expense of sm%, and vice versa. The relative importance of these two measures is the subject of some controversy, with the majority opinion being that reducing hm% is more important, but not *at all costs* with respect to increasing sm%. At TREC we attempted to sidestep the issue by reporting the logistic average (lam%) of the two scores, which rewards equally the same multiplicative factor in ham or spam misclassification odds. More formally:

$$\texttt{lam\%} = \texttt{logit}^{-1}\frac{\texttt{logit(hm\%)} + \texttt{logit(sm\%)}}{2}$$

where $\texttt{logit(x)} = \texttt{log(odds(x))}$

and $\texttt{odds(x)} = \frac{\texttt{x}}{\texttt{(100\% - x)}}$

Another way to sidestep the trade-off issue is to use the spamminess score to plot a Receiver Operating Characteristic (ROC) curve that represents all (hm%, sm%) pairs that could be achieved by the filter by changing a threshold parameter. Figure 1 shows the ROC curve for the best filter from each organization, as tested on the public corpus. In general, higher curves indicate superior performance regardless of the trade off between hm% and sm%, while curves that intersect indicate different relative performance depending on the relative importance of hm% and sm%. The solid curve at the top (ijsSPAM2full; Bratko's filter) shows sm% = 9.89% when hm% = 0.01%, sm% = 1.78% when hm% = 0.1%, and so on.

A useful summary measure of performance is the area under the ROC curve, ROCA, a number between 0 and 1 that indicates overall performance. In addition to the geometric interpretation implied by its name, this area represents a probability: the probability that the filter will give a random spam message a higher spamminess score than a random ham message. TREC reports (1-ROCA) as a percentage, consistent with the other summary measures which measure error rates rather than success rates.

TREC 2005's spam evaluation used three summary measures of performance: lam%, (1-ROCA)%, and sm% at hm% = 0.1. Each provides a reasonable estimate of overall filter performance; none definitively identifies the *best filter*.

RESULTS

The TREC spam evaluations generated a vast number of curves and statistics, which will appear in the TREC 2005 proceedings to be published early in 2006 (http://trec.nist.gov/pubs.html). We summarize the results with respect to the public corpus.

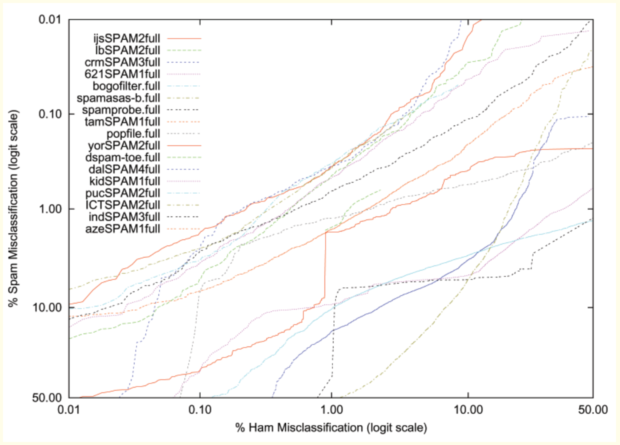


Figure 1: ROC curve for the best filter from each organization, as tested on the public corpus.

Run	Comment	Author
bogofilter	Bogofilter (open source)	David Relson (non-participant)
ijsSPAM2	PPM-D compression model	Andrej Bratko (Josef Stefan Institute)
spamprobe	SpamProbe (open source)	Brian Burton (non-participant)
spamasas-b	Spamassassin Bayes filter only (open source)	Justin Mason (non-participant)
crmSPAM3	CRM-114 (open source)	Bill Yerazunis (MERL)
621SPAM1	Spam Guru	Richard Segal (IBM)
lbSPAM2	dbacl (open source)	Laird Breyer
popfile	Popfile (open source)	John Graham-Cumming (non-participant)
dspam-toe	DSPAM (open source)	Jon Zdziarski (non-participant)
tamSPAM1	SpamBayes (open source)	Tony Meyer
yorSPAM2		Jimmy Huang (York University)
indSPAM3		Indiana University
kidSPAM1		Beijing U. of Posts & Telecom.
dalSPAM4		Dalhousie University
pucSPAM2		Egidio Terra (PUC Brazil)
ICTSPAM2		Chinese Academy of Sciences
azeSPAM1		U. Paris-Sud

Table 1: The selected test runs and their authors.

Table 1 associates each of the selected test runs (i.e. the best per organization) with its author. Only 12 of the filters were authored by official TREC 2005 participants; the other five were popular open-source spam filters, configured by the spam track organizers in consultation with their authors.

Run	Hm%	Sm%	Lam%
bogofilter	0.01	10.47	0.30
ijsSPAM2	0.23	0.95	0.47
spamprobe	0.15	2.11	0.57
spamasas-b	0.25	1.29	0.57
crmSPAM3	2.56	0.15	0.63
621SPAM1	2.38	0.20	0.69
lbSPAM2	0.51	0.93	0.69
popfile	0.92	1.26	0.94
dspam-toe	1.04	0.99	1.01
tamSPAM1	0.26	4.10	1.05
yorSPAM2	0.92	1.74	1.27
indSPAM3	1.09	7.66	2.93
kidSPAM1	0.91	9.40	2.99
dalSPAM4	2.69	4.50	3.49
pucSPAM2	3.35	5.00	4.10
ICTSPAM2	8.33	8.03	8.18
azeSPAM1	64.84	4.57	22.92

Table 2: The classification-based measures, ordered by lam%.

Table 2 shows the three classification-based measures (hm%, sm%, and lam%) for each filter, ordered by lam%. Note that hm% and sm% give nearly opposite rankings, indicating their heavy negative correlation and dependence on threshold setting. Table 3 shows the three summary measures: (1-ROCA)%, hm% at sm% = 0.1%, and lam% and the rank of each filter according to each of the measures. Note that while the rankings are not identical, they have a high positive correlation. The measures with respect to the other corpora vary somewhat but give the same general impression.

Run	(1-ROCA)%	*Rank*	Sm% @ Hm%=0.1	*Rank*	Lam%	*Rank*
ijsSPAM2	0.02	1	1.8	1	0.5	2
lbSPAM2	0.04	2	5.2	7	0.7	7
crmSPAM3	0.04	3	2.6	3	0.6	5
621SPAM1	0.04	4	3.6	6	0.7	6
bogofilter	0.05	5	3.4	5	0.3	1
spamasas-b	0.06	6	2.6	2	0.6	3
spamprobe	0.06	7	2.8	4	0.6	4
tamSPAM1	0.16	8	6.9	8	1.1	10
popfile	0.33	9	7.4	9	0.9	8
yorSPAM2	0.46	10	34.2	10	1.3	11
dspam-toe	0.77	11	88.8	15	1.0	9
dalSPAM4	1.37	12	76.6	13	3.5	14
kidSPAM1	1.46	13	34.9	11	3.0	13
pucSPAM2	1.97	14	51.3	12	4.1	15
ICTSPAM2	2.64	15	79.5	14	8.2	16
indSPAM3	2.82	16	97.4	16	2.9	12
azeSPAM1	28.89	17	99.5	17	22.9	17

Table 3: The summary measures and the rank of each filter according to those measures.

OBSERVATIONS

The most startling observation is that character-based compression models perform outstandingly well for spam filtering. Commonly used open-source filters perform well, but not nearly so well or nearly so poorly as reported elsewhere. We have reason to believe that reports on the performance of other filters are similarly unreliable; only standard evaluation will test their credence.

The main result from TREC is the toolkit and methods for filter evaluation. These may be used by anyone to perform further tests. The public corpus will be made available to all, subject to a usage agreement. The private corpora will remain in escrow so that new filters may be tested with them. Plans are already under way for TREC 2006, in which the same and new tests will be conducted on new filters and corpora. The new tests will include modelling of unreliable user feedback, use of external resources and other email processing applications.