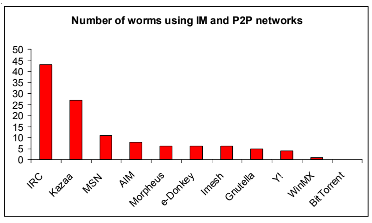
FEBRUARY 2005

virus BULLETIN

The International Publication on Computer Virus Prevention, Recognition and Removal

CONTENTS

IN THIS ISSUE



THE NEW BACKDOORS?

A security expert once told Dan Schrader that *Yahoo! Instant Messenger* was nothing more than a port scanner with an application attached – he laughed, but fears that the expert was in fact right.

page 9

GENERICALLY SPEAKING

Since the release of Code Red, the online world has struggled to protect against – and in many cases recover from – 'blended' threats, including 2003's Slammer and Blaster. Carey Nachenberg looks at a technology that holds the promise of stopping many of these threats proactively: generic exploit blocking.

page 6

RECORD BREAKERS

A *VB* record-breaking 28 products line up for this month's *Windows NT* comparative review.

page 12

vbSpam supplement

This month: anti-spam news & events; combining multiple spam classifiers; ASRG summary.

ISSN 0956-9979

VIRUS BULLETIN COMMENT

'It seems clear that Microsoft is poised to plough ahead with its security plans.'

Mary Landesman
Antivirus Guide About.com, USA

MICROSOFT'S GIANT INITIATIVE

In April 2003, *Microsoft* quietly recruited beta testers for PC Satisfaction – a managed service aimed at consumers and small businesses. *Microsoft* described the initiative as a 'Real People, Real Data' customer research program.

Having hinted at a spyware interest for months, *Microsoft* announced its acquisition of *Giant Software* in December 2004. The announcement was quickly followed by a public beta of *Microsoft AntiSpyware*.

We pitted *Microsoft AntiSpyware* (beta1) against some of today's most prevalently tagged adware and spyware. We also tested *Lavasoft*'s *Ad-Aware SE Personal v1.05*, and *Spybot Search & Destroy v1.3*. All of the products were updated to the most current definitions available.

Microsoft AntiSpyware averaged a 97 per cent overall detection rate. 100 per cent of the start and search page hijackers and 100 per cent of the BHOs (Browser Helper Objects) were detected and corrected. Running processes and executables loading via the Registry run keys were detected at 96 per cent and 95 per cent respectively. Both *Spybot* and *Ad-Aware* averaged a 63 per cent overall detection rate:

	MS beta1	*Spybot*	*Ad-Adware*
Overall detection	97%	63%	63%
Start & search page hijackers	100%	75%	25%
BHOs	100%	57%	57%
Running processes	96%	63%	70%
Executables loading via Registry run keys	95%	64%	73%

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*

In addition to strong detection, *Microsoft AntiSpyware* includes fairly robust prevention – warning when attempts are made to change browser pages, to add sites to the Trusted Sites zone, or to modify the HOSTS file. But strong detection rates do not necessarily translate into a product that is enterprise-ready. Like most of its competitors, *Microsoft AntiSpyware* lacks centralized management features. And, according to *Microsoft*, its script-blocking feature can 'potentially prompt the end user to make decisions to allow or block administrative actions that originate from a central management tool'. Another conflict exists when the product is installed alongside *Windows XP Media Center Edition 2005* – the Media Center is unable to establish remote connections.

Microsoft AntiSpyware also includes features that allow users to modify or delete registry startup items and processes, as well as modify existing HOSTS file entries – controls which many system administrators might prefer their users did not have.

Reaction to the *Microsoft* spyware initiative has been mixed. Some have applauded *Microsoft*'s decision, with the caveat that the protection be free. Others argue that providing free security software constitutes unfair competition. Still others point out that it appears as if *Microsoft* is betting against itself. With *Longhorn* on the horizon and *XP SP2* barely dry, they view these acquisitions as public admissions of a lack of faith in its ability to build a more secure OS.

Regardless of praise or criticism, it seems clear that *Microsoft* is poised to plough ahead with its security plans. Hot on the heels of the *Microsoft AntiSpyware* beta, *Microsoft* debuted its *Malicious Software Removal Tool* – which is slated to be updated monthly following the same 'second-Tuesday' cycle as its security patches. According to the EULA, once the tool has been installed, 'if you receive updates via Automatic Updates, you will only be able to stop receiving updates to this software if you turn off Automatic Updates entirely.' The EULA also notes, 'Microsoft may collect and publish aggregated data about the use of the software.'

Perhaps this is the first peek at what *Microsoft* product manager Nicolas Mirail allegedly leaked to *ZDNet* in July 2004. Mirail reportedly said that, 'the Microsoft antivirus software will utilize two different means of detecting destructive files, the first of which will reference a regularly updated list of known viruses to check for potential infections. The second antivirus tool will analyse computer systems to assess whether they have been hit by a virus in the past and attempt to give end users an idea of how at risk their computers might be for future problems.' Real People, Real Data, indeed.

NEWS

CALL FOR PAPERS: VB2005 DUBLIN

The deadline for submission of abstracts for VB2005 is approaching rapidly. All submissions must be received by **10 March 2005**.

VB2005, the Fifteenth Virus Bulletin International Conference, will take place 5–7 October 2005 at The Burlington, Dublin, Ireland.

Submissions are invited on all subjects relevant to the anti-virus and anti-spam arenas. A list of suggested topics elicited from attendees at VB2004 can be found at http://www.virusbtn.com/conference/vb2005. Papers on these and any other AV and spam-related subjects will be considered. *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.

Abstracts of approximately 200 words must reach the Editor of *Virus Bulletin* no later than Thursday 10 March 2005. Submissions received after this date will not be considered. Abstracts should be sent as RTF or plain text files to editor@virusbtn.com. Following the close of the call for papers all submissions will be anonymised before being reviewed by a selection committee; authors will be notified of the status of their paper by email. Further details of the paper submission and selection process are available at http://www.virusbtn.com/conference/vb2005/.

TROJAN AUTHOR ARREST

The Spanish Civil Guard announced the arrest last month of a man suspected of writing and distributing a Trojan capable of covertly spying on recipients via their webcams, as well as stealing banking information. It is alleged that the man (identified only as 'J.A.S.' and who, at 37 years old, is somewhat more advanced in years than the 'typical' virus writer) distributed the Trojan via file-sharing networks. The Civil Guard said it had been investigating the case in 'Operation Tic-Tac' since July 2004, when a computer user used the Civil Guard's website to report a suspicious file on his computer. Following investigation, the Trojan was traced to J.A.S., whose computer was seized and found to contain 'hundreds of photographs and recordings'.

Meanwhile, the UK's Metropolitan Police Force is launching a series of computer crime prevention and response seminars for businesses aimed at improving companies' incident response plans for managing internal and joint police investigations. Three seminars will be held during 2005 – one for corporate managers, one for security professionals and one for corporate investigators.

Prevalence Table – December 2004

Virus	Type	Incidents	Reports
Win32/Netsky	File	117,348	50.71%
Win32/Sober	File	66,012	28.53%
Win32/Zafi	File	20,446	8.84%
Win32/Bagle	File	20,341	8.79%
Win32/Bagz	File	1,997	0.86%
Win32/Mydoom	File	937	0.40%
Win32/Mabutu	File	749	0.32%
Win32/Dumaru	File	639	0.28%
Win32/Funlove	File	495	0.21%
Win32/Klez	File	389	0.17%
Win32/Lovgate	File	261	0.11%
Win32/Bugbear	File	222	0.10%
Win32/Valla	File	146	0.06%
Win32/Swen	File	134	0.06%
Win32/Mimail	File	106	0.05%
Win95/Spaces	File	106	0.05%
Win32/Kriz	File	92	0.04%
Redlof	Script	88	0.04%
Win32/MyWife	File	84	0.04%
Win32/Mota	File	77	0.03%
Win32/Pate	File	77	0.03%
Win32/Fizzer	File	62	0.03%
Win32/Yaha	File	58	0.03%
Win32/Mugly	File	44	0.02%
Win32/Hybris	File	42	0.02%
Win32/Magistr	File	32	0.01%
Win32/Sobig	File	31	0.01%
Win95/Tenrobot	File	27	0.01%
Win32/Elkern	File	26	0.01%
Win32/BadTrans	File	23	0.01%
WYX	Boot	23	0.01%
Win32/Plexus	File	18	0.01%
Others[1]		271	0.12%
Total		231,403	100%

[1]The Prevalence Table includes a total of 271 reports across 59 further viruses. Readers are reminded that a complete listing is posted at http://www.virusbtn.com/Prevalence/.

FEATURE 1

THE THREE FACES OF VBA: PART 2

Dr Vesselin Bontchev
FRISK Software International

Last month (see VB*, January 2005, p.12) Vesselin Bontchev shared some of his specialist knowledge of macro viruses in an introduction to the description of a particular problem: the fact that macros written in VBA have three different forms, any of which can be the one that is executed. In this part of the article, Vesselin provides a detailed explanation of the problems related to this fact.*

THE EXECUTABLE SOURCE

If we consider the algorithm outlined in part 1 of this article carefully, it becomes obvious that in some circumstances (i.e. when a document containing a VBA macro is opened with a version of *Office* that uses a version of VBA that is different from the one that has created the macro), it is the contents of the source code that will control what actually is executed – not the contents of the p-code. This causes two different problems for anti-virus developers.

First, it means that those anti-virus programs whose designers followed *Microsoft*'s advice to scan the source code will have achieved a level of 'upward compatibility', similar to that of *Office* itself. For example, as the p-code of an *Office 97* virus is upconverted to *Office 2000* form, such anti-virus programs will continue to be able to detect the virus without any problems. Meanwhile, the anti-virus programs that look exclusively at the p-code will be forced either to treat the upconverted form as a different virus (even if they report it under the same name), or to resort to on-the-fly p-code conversion, using the methods outlined in [3].

Of course, this does not mean that scanning just the source code is the only way to proceed. Anti-virus programs that do so are vulnerable to a different kind of problem. For instance, a virus writer could wipe (or otherwise modify) the source area of his virus, either manually (e.g. with a hex editor) or programmatically. As a result, the anti-virus programs that scan only the source code area will be unable to identify it. Of course, it would also mean that the virus would run properly only under the same version of VBA as the one that created it – but virus writers are rarely too concerned about compatibility issues.

The second problem is related to disinfection. The early versions of some anti-virus programs that looked only at the p-code area also disinfected VBA viruses by wiping *only* the p-code area and leaving the source code area intact. This had some unpleasant consequences. First, it meant that all anti-virus programs that looked only at the source code continued to report the disinfected documents as containing the virus. Second, it meant that if the 'disinfected' document was opened under a different version of *Office*, the virus would suddenly reappear in it, seemingly from nowhere. This is why today's anti-virus programs (even those that still look mainly at the p-code area) either remove completely the module streams that belong to the virus, or wipe *both* the p-code and the source code areas in them.

These problems forced me to make some modifications to our macro virus scanner. It still looks mainly at the p-code area – because, most of the time, this is precisely what is executed. However, I introduced a new heuristic. Now, if the p-code area is missing or empty, but a sufficiently large source code area is present, our scanner would report the document as 'possibly mis-disinfected' – hinting that a scanner may not have bothered to wipe the source when disinfecting a virus.

Clearly, virus writers are aware of the problems caused by the fact that sometimes the p-code controls the execution and sometimes the source code does.

For instance, the W97M/Class.EZ virus uses this fact to avoid being detected by scanners that look only at the p-code area. When infecting documents, it damages the pointers that are normally used to locate the p-code area in the module stream containing the virus. As a result, the scanners that look for a p-code area to scan are unable to find it. However, the virus also modifies one additional byte in the stream, indicating that the VBA project has been created by an impossibly high version of VBA – i.e. one which is different from all the existing ones. This forces *Word* (any version of it that supports VBA) to re-compile and run the contents of the source code area and, therefore, to execute the virus – despite the fact that it wouldn't be able to locate its p-code area.

LIKE A JINI FROM A BOTTLE

The fact that the execodes (as opposed to the p-code or the source code) can also control what is executed under some circumstances (i.e. when they exist and when the VBA project containing them is opened by exactly the same version of VBA as the one that has created it) causes its own set of problems for the anti-virus programs.

When I first became aware of this, I performed some experiments and reached the conclusion that, while this fact could be used to conceal a Trojan horse (e.g. by wiping or otherwise modifying its p-code and source code area, in order to make it undetectable with an anti-virus program that looks only at these areas), it could not be used to conceal a virus. My reasoning was that a macro virus replicates basically by saying 'copy my module(s) from here to there'. However, if the module streams do not exist, or are otherwise corrupted, they will be unable to copy themselves. As a result, the macro malware will be unable to copy itself – i.e. it will no longer be a virus. The malicious payload of a Trojan could still be active, however – even if present only in execodes form.

Unfortunately, my reasoning was proven wrong. Worst of all, it was proven wrong by a virus writer who was not even trying to do so – he was trying to solve a completely different problem.

The contents of a VBA project can be 'protected' with a password. I have put the word protected in quotation marks, because the contents of the modules are not encrypted – and it is, in fact, quite trivial to extract them with a third-party program. The 'protection' is basically a flag that tells VBA that the VBA Editor is not allowed to display the contents of the VBA project, unless the user supplies the proper password. (In VBA version 5.0 the password is stored, in a trivially encoded form, in one of the information streams, while in the later versions of VBA only a cryptographic hash of it is stored there.)

However, there is one additional restriction imposed on protected VBA projects – namely, their modules cannot be copied elsewhere. This restriction is quite natural – if it did not exist, people would be able to circumvent the protection simply by writing a VBA program that copies the modules of the protected VBA project to a different VBA project that is not protected. However, the implementation of this restriction also means that a module residing in a protected VBA project is not allowed to copy *itself* or any other module in that project. In other words, a VBA virus cannot reside in a protected VBA project, or it will be unable to copy itself in the usual way.

Of course, if the VBA project of the virus is not protected, anyone can examine its contents with the VBA Editor. Some virus writers would rather avoid this, so they sought ways to circumvent the restriction.

One possible way to circumvent it is to create a virus that does not try to infect other documents, but which replicates by always sending one and the same document – that in which it resides. This sending can be performed by email, by IRC, by crawling the network shares, and so on. But what if the virus writer wants his virus to be able to infect other documents?

Enter the X97M/Jini.A virus.

The author of this virus had a really 'bright' idea. He probably thought '*If a virus that resides in a protected VBA project cannot copy itself to other documents, can't it copy other documents to itself, instead?*'. And this is precisely what the virus does. It infects *Excel* workbooks – but it does not copy itself to them. Instead, it deletes all sheets from the workbook where it resides, then copies all sheets from the target workbook to the workbook where it resides, then closes and deletes the target workbook, and finally saves the workbook where it resides under the file name of the target workbook. All this hocus-pocus is needed just to be able to circumvent the restriction that modules residing in protected VBA projects are not allowed to copy themselves elsewhere. However, the bizarre infection method used by this virus has some unintended consequences.

Obviously, at some point, a file infected with this particular virus had been disinfected by a scanner that knew the virus, but which did not bother to remove the __SRP_* streams when disinfecting it. As a result, the module stream of the virus (with the p-code and the source code areas residing in it) was removed from the infected file – but the streams containing the execodes were left intact.

Had this happened to a 'normal' virus, the result would have been that the virus was unable to replicate – because execodes in the document would have tried to copy non-existent module streams. But X97M/Jini.A does not try to copy any streams. Instead, it copies the data sheets of the target to the file where it resides and overwrites the target with that file. Since that file also contains the execodes, it means that the execodes would be able to replicate on their own – even without the presence of the module stream that contains the p-code and the source code.

So, thanks to *Microsoft*'s reluctance to provide relevant information and to the sloppiness of some unknown anti-virus producer, we ended up with a macro virus that was undetectable *both* to the anti-virus programs that looked at the p-code and to those that looked at the source code.

How could the resulting problem be solved? My initial reaction was something similar to the heuristic described earlier: if neither p-code, nor source code was found, but any execodes were present, the document would be reported as 'possibly mis-disinfected'. Unfortunately, this 'simple' solution turned out being problematic.

Apparently, if an *Excel* spreadsheet contains complex web controls (like radio buttons, checkboxes, dropdown listboxes and so on), some execodes are generated in order to handle them – even though there isn't any VBA program in the document. Since, in such cases, there is neither p-code nor source code but there *are* some execodes, my heuristic triggered, causing false positives. And such false positives were practically impossible to 'fix' – because my way of fixing macro false positives is to create a special entry in our database, corresponding to a macro from the macro package that was causing the false positive. The scanning engine, when finding a macro described by this entry, simply suppresses the report from the heuristics. But in these cases there *were* no macros – so I could not create any special entries for them.

Eventually, I solved the problem by reverse-engineering the format of the execode streams to a certain degree, implementing a parser for them, and modifying our scanner so that it was able to handle virus definition entries that describe not just p-code areas of macros but also execode areas.

Unfortunately, I am not sufficiently confident in the correctness of my reverse-engineering to document its results here. As mentioned in the first instalment of this article, even locating the beginning of the actual execodes in the execode streams is based on a set of heuristics – not on an algorithm that is guaranteed to work reliably every time. But the heuristics seem to work reliably in most test cases I have been able to construct, and they definitely work reliably to detect all instances of the single execodes-only virus in existence – so, for my purposes, they are considered good enough. Of course, I will continue to research this issue, to improve my algorithms, and to press *Microsoft* for more information.

SUMMARY AND CONCLUSION

As we saw from the previous sections, beginning with *Office 97*, VBA macros can exist in three completely different forms – any of which can dictate what would be actually executed at runtime, depending on the particular circumstances. In some obscure cases, the contents of these three forms might be expressing different algorithms – either intentionally, because of some trick used by the virus writer, or unintentionally, due to the sloppiness of an anti-virus program.

It is therefore imperative that anti-virus programs are capable of handling malware that exists in any of these three forms – even if any of the other two forms are not present or not malicious. At the same time, it is imperative that all three forms are removed when a document containing macro malware is being disinfected.

REFERENCES

[1] Igor Muttik, ‘A Portrait of Jini’, *Virus Bulletin,* October 2000, p.7.

[2] http://www.etree.com/tech/freestuff/edoc/eDoc.exe.

[3] Vesselin Bontchev, ‘Solving the VBA upconversion problem’, *Proc. 10th Int. Virus Bull. Conf.*, pp.273–299.

[4] Costin Raiu, personal communication.

FEATURE 2

GENERIC EXPLOIT BLOCKING

Carey Nachenberg
Symantec, USA

17 July 2001 was a big day for anti-virus vendors (as well as for computer users). Code Red, the first blockbuster network worm to be released since the Morris worm of 1988, spread swiftly across the Internet, compromising hundreds of thousands of machines in a brief eight hours. Unlike most of the worms around at the time (e.g. email and drive-sharing worms) which spread in a file-based form, Code Red (see *VB*, August 2001, p.5) did not spread as an executable file; it existed only in infected computers’ RAM chips and as light pulses and electrons as it travelled across the Internet. Moreover, Code Red was entirely autonomous, infecting machines without the need for users to open attachments or double click a file.

Nightmares became realities that day, forcing anti-virus providers to come to terms with two game-changing issues. First, we recognized that stand-alone, file-based anti-virus software could not detect such threats; there was no file to scan! The definition of anti-virus would have to evolve to include technologies that could scan and block network-based threats. Second, we learned that such autonomous worms could spread so rapidly that they broke the traditional reactive signature-based anti-virus model. By the time a security provider could create and deploy a signature for a new worm, most susceptible machines were already infected, relegating anti-virus software to infection removal rather than infection prevention. The lack of solutions to these problems has reduced the utility of traditional anti-virus software and has increased costs for corporations and end-users.

EMERGING TECHNOLOGIES

Two complementary classes of technology have emerged to address the deficiencies of traditional anti-virus software and tackle such network-based worms: confinement technologies and preventative technologies.

Confinement technologies (often called ‘behaviour blockers’) work by monitoring software as it runs on a protected computer. If the confinement software detects a program exhibiting suspicious behaviour (i.e. sending lots of network packets to other computers that have never been contacted before), the confinement software can block or slow down the suspicious application. Confinement-based approaches can work only *after* a computer has been infected, attempting to inhibit the infection from spreading to other machines. This is analogous to a building security system that, upon detecting an intruder inside the building, locks all internal doors to surround the intruder, preventing him from roaming through or exiting from the building. Examples include buffer-overflow blockers, memory-scanners, API blockers and rate-limiting systems.

In contrast, preventative technologies block network-based threats from infecting a protected computer in the first place, much like a guard prevents a burglar from entering a building. Examples include personal and enterprise firewalls, and network intrusion prevention systems. All of these systems inspect network packets and can block a malicious transmission before it reaches and infects a susceptible computer, preventing infection. Generic exploit blocking is such a preventative technology.

LOCKS, KEYS AND SOFTWARE VULNERABILITIES

A software vulnerability is a flaw in an application that allows an attacker to compromise the application and gain control over a computer system. Arguably, the most severe type of software vulnerability is the network-accessible vulnerability. In this type of vulnerability the application expects, but fails to actually enforce, that the incoming packet data is in a well-defined format and of a specific, limited length. An attacker compromises the vulnerable software by sending incoming packets containing data that violates these expectations – the attacker sends packets that contain data that is too long and/or contain misencoded data. Since the vulnerable application fails to check for misencoding, it is tricked into performing a malicious action on behalf of the attacker. A set of malicious packets that is directed at a vulnerable application is called an exploit.

Just as a padlock has a set of internal pins that limit the shape of a key that can open it, every software vulnerability has a specific internal structure that can only be attacked by a properly crafted set of exploit packets. If a key doesn't have the right shape, it can't open a padlock, and similarly, if an attacker sends improperly crafted exploit packets to a vulnerable piece of software, those packets will fail to compromise the vulnerable software.

As a consequence of the lock-and-key relationship between a vulnerability and an exploit on that vulnerability, any exploit packets targeting a given vulnerability *must* be appropriately shaped to the vulnerability to compromise it. Assuming security researchers can exactly characterize the vulnerability's structure and internal shape, then they can also characterize the exact required shape of exploit packets. This realization gives security researchers a powerful tool to generically detect and block new attacks.

For example, imagine that a particular model of padlock is widely used around the country. Furthermore, assume that it is found that the padlock is vulnerable to picking (i.e. it has a vulnerability). By x-raying such a vulnerable lock, a locksmith could determine that, in order to pick the lock, the pick must have a shaft that is exactly 4cm long, 0.7cm high, with 0.4cm-high grooves spaced at 2cm, 3.7cm, and 5.2cm on the pick. Once the locksmith has completed his analysis, he could write these specifications down, without ever having seen an actual pick for this class of vulnerable lock. The locksmith could hand these specifications of the lock pick to the police and instruct them to prevent anyone from approaching a vulnerable lock if they hold in their hand a pick matching the specifications.

Upon learning of the vulnerable lock, criminals may attempt to produce many different types of picks to fit the lock: some picks may be made of aluminium, others formed of plastic; some picks may have a large square handle, and other picks may have a small round handle. However, all of these picks, in order to work, must have a shaft meeting the requirements defined above (4cm long, 0.7cm high, etc.). Regardless of the colour of the pick, the size or shape of its handle, or what material it is made from, the police officer can identify any viable pick by the length and width of its shaft and the pattern of grooves. As long as the pick meets the minimum requirements, it can be identified and blocked.

In a similar fashion, given a new software vulnerability for which no worm or other attacks yet exist, a security researcher can analyse the vulnerable software and determine the inherent structure of its vulnerability and the complementary required shape of exploit packets. The researcher can then create a vulnerability signature, not unlike a traditional anti-virus signature, to recognize all exploit packets that have this complementary shape. Such a signature can be created well before any attacks (e.g. hacking tools or worms) are released that attack the vulnerability, since the shape of the vulnerability is inherent to the vulnerable software.

Vulnerability signatures are described in terms of attributes such as:

1. The required network port and protocol to which exploit packets must be sent to cause the attack, e.g. 'The packet must be sent to port 1434 via the UDP protocol.'
2. The required contents of the exploit packets, e.g. 'The packet must have a value of 4 in the first byte of the packet to fit the vulnerability.'
3. The length of particular data packets or of data structures within those packets, e.g. 'The packet must be more than 60 bytes long.'

Moreover, the vulnerability signature must specify only those structural/shape characteristics that are absolutely required to attack a given vulnerability. Consider our earlier lock pick example. In this example, the vulnerability signature detected picks with a shaft that is exactly 4cm long, 0.7cm high, with 0.4cm-high grooves spaced at 2cm, 3.7cm, and 5.2cm. However, this signature did not specify constraints such as the pick's material, the shape of its handle, etc. While these attributes would be useful to detect a *specific* lock pick, they may vary across many different lock picks. Only the fundamental dimensions of the shaft of the pick may be included in the signature.

How do vulnerability signatures apply to network-accessible computer worms? Network worms like Code Red, Slammer, and others use network-accessible vulnerabilities to break into computers and replicate. Such a worm first identifies a new target machine and then sends a set of properly shaped exploit packets to the target machine. These exploit packets enable the worm to compromise the vulnerable software, take control of the computer, and continue the replication process. Thus, if we can generically block exploit packets for a particular vulnerability, we can consequently block all worms that attempt to spread via that vulnerability. By creating and deploying a vulnerability signature to desktop firewalls, enterprise firewalls, and intrusion prevention systems, we can generically detect and block the worm's exploit packets before they can reach and infect new machines.

Vulnerability signatures may appear at first sight to be similar to traditional anti-virus heuristics. In fact, vulnerability signatures are much more robust than traditional heuristics. Traditional anti-virus heuristics work by scanning a file or network stream for a series of suspicious-looking instructions or data sequences, for example instructions that appear to modify files, format the hard drive, or establish new Internet connections. Ideally, the heuristics are well designed to detect a wide variety of threats. However, there is no guarantee that a particular virus or worm actually employs these instruction sequences. Furthermore, the sequences may be hidden via encryption, compression, instruction reordering, or perhaps the virus author has decided to 'tweak' his virus until it no longer uses logic that is detected by existing heuristics.

In contrast, a properly written vulnerability signature simply cannot be bypassed. If the worm's author tweaks his worm's exploit packets until they are no longer detectable by a vulnerability signature, then the worm's exploit packets will fundamentally have the wrong shape and be unable to fit with and exploit the targeted vulnerability.

This is an important result, for it means that in those cases where we can properly characterize the shape of a vulnerability, we can proactively create a signature that detects and blocks all future attacks on that vulnerability.

Unfortunately this approach applies only to network-based worms; vulnerability signatures cannot be applied to traditional parasitic computer viruses or email-based worms. Traditional parasitic computer viruses spread by attaching their logic to the existing logic of a host executable file. Such traditional viruses can attach or inject themselves virtually anywhere within a target executable file. They do not have to have a particular shape to infect a particular executable file and there is no similar analogy of lock and key with a traditional virus.

Similarly, email worms don't need to have any particular shape to spread across the Internet; all the email worm needs to do is insert itself to outgoing emails as a standard executable file attachment and it can spread. However, network worms like Code Red, Slammer, Sasser, Blaster and dozens of others must send properly shaped exploit packets to break into new machines and therefore all of them could have been blocked with this technology, had it been deployed at the time of infection.

GENERIC EXPLOIT BLOCKING

We call this approach of creating vulnerability signatures generic exploit blocking (GEB), since it can be used to generically block *all* future exploits against a given vulnerability. Others in the industry call it shielding or virtual patching.

While extremely promising, the generic exploit blocking approach is by no means foolproof. There are several potential problems with the generic exploit blocking approach. First, certain vulnerabilities are so complex that it is difficult to characterize their shape accurately. In such a situation, we can create a vulnerability signature, but it may end up missing certain difficult-to-characterize attack shapes.

A second potential problem with this approach is that a particular vulnerability may be susceptible to numerous, different-shaped attacks. Referring back to our analogy, if any shaft with any set of grooves can pick our vulnerable lock, then a shape detection sketch for such a lock will likely be so general that it recognizes both malicious keys/picks for our susceptible lock as well as legitimate keys for other locks. Such a signature will result in too many false positives, rendering it useless.

A third problem is that, while we may be able to characterize exactly the shape of a given vulnerability, it may be computationally expensive to search for this shape in network packets at ultra-high data rates. As most computer users have experienced first hand, anti-virus scanning is computationally expensive and the algorithms employed by generic exploit blocking systems are not necessarily less expensive. As data rates surpass gigabit and

10-gigabit speeds, this approach may be impossible without specially tailored pattern-matching hardware.

Finally, the generic exploit blocking approach can be used only if the targeted vulnerability has been identified before an attack (e.g. a worm) takes place. Depending on the nature of the new vulnerability, it may take a security researcher days or weeks from its discovery to analyse it completely and produce a viable vulnerability signature. If a worm is released that attacks the vulnerability before the signature can be written, then this approach will fail to stop the worm. Similarly, day-zero attacks, where a worm exploits a heretofore undiscovered vulnerability, cannot be protected by the generic exploit blocking approach. Thus, while extremely effective, this approach is by no means perfect.

Even given these caveats, initial research leads us to believe that the generic exploit blocking approach is feasible for more than 50 per cent of all network-based vulnerabilities. Had this technology been deployed and proper vulnerability signatures been written and distributed to customers, virtually all of the high-profile network worms we have seen since the emergence of Code Red could have been stopped proactively (with the exception of the Witty worm which was, in essence, a day-zero worm, and hence not an applicable target for generic exploit blocking).

Even more extraordinary, unlike traditional anti-virus scanning algorithms which must constantly be updated by security vendors to keep up with the latest advances in virus obfuscation (e.g. packing, polymorphism and metamorphism), a generic exploit blocking system does not face these hurdles. For a network worm to exploit a vulnerability, the worm author *must* tailor the shape of the worm's exploit packets to the vulnerability. In other words, it is the shape of the application vulnerability that dictates the telltale shape of the worm, *not* the worm's author. Consequently, obfuscation advances by virus writers in no way compromise the effectiveness of a GEB system. This drastically reduces the need to update constantly and co-evolve the GEB system along with the latest threats. While viruses and worms evolve, vulnerability shapes do not, therefore limiting the need to update the GEB scanning technology.

Generic exploit blocking is appropriate in both enterprise and consumer environments and can be used in any device that has the ability to scan network packets. Since the release of Code Red, the online world has struggled to protect against – and, in too many cases, recover from – other blended threats, including 2003's Slammer and Blaster. Generic exploit blocking technology holds the promise to stop many of these worms proactively, before they can infect susceptible machines, reducing the need for middle-of-the-night patch deployment and costly clean-up of infected computers.

FEATURE 3

THE NEW BACKDOORS: IM AND P2P NETWORKS

Dan Schrader
FaceTime Communications, USA

Imagine, if you will, the world's largest technology companies competing to give away the tools of hackers – port scanners, adware and software for untraceable communications. Imagine the uproar if *Microsoft*, *Yahoo!* or *AOL* were to offer, free of charge, software for poking holes through your firewalls and shredding your perimeter defences. Well, the silence is deafening; there has been no uproar, even though the companies named above, and many more, are doing just that.

A security expert once told me that *Yahoo! Instant Messenger* was nothing more than a port scanner with an application attached. I laughed – but he was right. *Google* will likely soon join *Microsoft*, *Yahoo!* and *AOL* in offering free instant messaging. It is likely that *Google*'s software will match the other vendors' offerings in being 'port agile', being able to revert to HTTP tunnelling if no ports are available, in providing file transfer capability, and in offering the freedom of anonymity (in other words, lack of accountability through authentication). In short, these instant messaging (IM) networks and their peer-to-peer (P2P) cousins are 'dark' networks, designed to evade firewalls and designed with barely a nod to the security needs of the networks in which they will often run.

THE GROWING THREAT

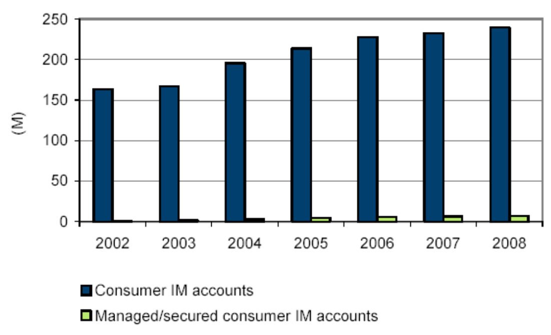
Traditionally, the information security community has done a poor job of identifying and securing communication channels pro-actively. It took a decade of virus growth before vendors started shipping computers that didn't boot first from the floppy drive. Security experts warned for years of the threat of email-borne viruses before the Melissa virus struck, yet few were ready when it did. The Internet itself, the network without authentication, is a monument to faith in human nature trumping common sense.

Most IT organizations show similar credulity when faced with IM and P2P. While a small number of organizations have implemented management systems for these 'real-time' networks, and a slightly larger number have posted written policies restricting their use, most organizations have ignored IM and P2P, hoping these networks will just go away. However, the opposite is occurring.

Instant messaging and peer-to-peer traffic is growing at a staggering rate. Some authorities estimate that as many as 300 million people use instant messaging. They are not all teenagers. *Osterman Research* recently found that IM is in use in 90 per cent of all commercial and non-commercial enterprises. They went on to estimate that use of IM 'will grow to about 80 per cent of all email users by 2007' (Managing IM and P2P Threats in the Enterprise, *Osterman Research*, September 2004).

IDC recently broke down IM use into enterprise and consumer use. *IDC*'s report estimated that by the end of 2004, approximately 30 million people will have access to an enterprise IM system such as *IBM*'s *Sametime* or *Microsoft*'s *LCS* (Worldwide Enterprise Instant Messaging Applications 2004–2008 Forecast and 2003 Vendor Shares, *IDC*, October 2004).

Worldwide Total Consumer Instant Messaging Accounts: Managed/Secured Versus Unmanaged, 2002–2008



Meanwhile, *IDC* believes that consumer use of IM dwarfs enterprise roll-outs, with nearly 200 million IM clients installed by the end of 2004.

In addition to traditional 'pure' IM clients such as *MSN*, *Yahoo!* and *AIM*, instant messaging functionality is increasingly being added to other applications using 'real-time' communications networks such as voice over IP (VoIP) and peer-to-peer clients. *Skype*, for example, which was first shipped in 2004, will soon announce its 40-millionth download.

Not only has the total growth of IM use far outstripped growth in new email or Internet accounts, but the numbers of buddies per user has grown, creating a rich environment in which worms can spread. Meanwhile, multi-network clients such as *Trillion* and *PalTalk* are becoming increasingly popular, allowing worms to spread across IM networks. These multi-network clients allow users to have one interface for communicating with buddies on *AIM*, *MSN* and *Yahoo!* The bottom line: an aggressive IM worm could easily spread to tens of millions of computers in remarkably short order.

Many analysts term IM and P2P 'disruptive technologies' because they are driven by end-user demand rather than by IT fiat, and because they change the way people work and communicate. Perhaps more to the point, IM and P2P are disruptive because they will, eventually, disrupt your network security in some fairly predictable ways.

THE IM AND P2P SECURITY ISSUES

IM and P2P pose a number of security-related issues. While (at least to readers of this publication) malicious code may be the most obvious, those who have purchased IM management and security products have done so primarily to comply with regulatory issues. In the US, there is a maze of regulations, guidance and the like regarding electronic messaging retention, security and privacy. Some of those regulations include the following:

Sector	**Regulatory Mandate**
Banking	FDIC IM & P2P Guidance 7-21-04
Invest. Banking	SEC 204-2, 31 a/b
Broker/Dealer	SEC Rule 17a-4, NASD 2210/3010
Insurance	FDIC, State, SEC 17a-4
Life Sciences/Pharmaceuticals	US FDA 21 CFR 11
Health Care	HIPAA
Energy	FERC Record-keeping
Gov	DoD 5015.2, FOIA, GRS 20, NARA

Surprisingly, the security issues surrounding IM and P2P have escaped the notice of most IT professionals – and of most security vendors. Those security threats can be split into four areas as follows:

Vulnerabilities – weaknesses in the client software that may be exploited to expose information, deny service, change data or provide a launch pad for further attacks. Until people learn how to write perfect software, vulnerabilities will remain a fact of life. Security holes requiring patching have been found in every major IM client. It is difficult enough to maintain and patch software that you have authorized, but with end users downloading and installing any of over two dozen different IM and P2P clients, patch management becomes impossible.

Loss of confidentiality – IM networks typically send messages by plain text over public network. Encryption is rarely used in the IM world.

Loss of control over intellectual property. IM and P2P file transfers are not limited to trading music. Source code, research reports, product information or just about any type of information can be exposed by IM use.

Malware distribution – IM networks are a natural ecosystem for viruses and other malicious code. Like most consumer products, and like the Internet itself, IM networks have been developed with little thought to information security. In fact, virus writers have every reason to be drawn to IM.

IM AND VIRUS WRITERS

Virus writers use email to spread their code for a number of reasons, among them, because email is ubiquitous, it is fast and because SMTP engines are freely available among the virus underground. However, from the perspective of the malicious code author, IM networks offer a number of benefits that make them quite attractive as a distribution and communications medium. If I were asked to compile a 'top ten' list of the reasons why virus authors like IM and P2P, it would have to be a 'top twelve':

12. IM clients do the hard work for them. By bypassing blocked ports automatically and by tunnelling through HTTP traffic, IM clients create a reliable and difficult-to-block channel for distributing malicious code.
11. Virus writers know that gateway-based scanners will not block IM and P2P traffic. Hackers are well aware that email scanners limit the spread of their 'wares'. However, email and firewall-based anti-virus products do not scan IM traffic.
10. Multi-network clients such as *Trillian* and *Microsoft LCS*'s new connectivity options dramatically increase the number of machines and the degree of interconnectivity available to IM worms.
8. IM provides an easy and reliable channel the virus writer can use to retrieve information from infected machines.
7. IM renders images directly to the screen. Since images are not saved to the local machine first as a file, desktop on-access scanners will not detect viruses embedded in JPEG or bitmapped images. (This issue remains more theoretical than real, there are no image-based viruses in the wild, but that is not to say that there never will be.)
6. IM networks allow remote control of backdoors without opening a port. Through IM and P2P, backdoors and other malware can be managed remotely – even in the presence of corporate and desktop firewalls. As *Symantec* recently pointed out, 'if the backdoor Trojan horse operates via the instant messaging client, it does not open a new port and thus, is not blocked by traditional desktop firewall products.' ('Threats to Instant Messaging', Neal Hindocha, Symantec Security Response Team, 2004.)
5. IM can have links to infected web pages – which can have malicious code based on buffer overflow conditions, cross-site scripting or other browser vulnerabilities.
4. If you can install an IM client, you probably have local administrative rights.
3. Most companies rely on 'tootsie roll' security – hard and crunchy on the outside, soft and chewy on the inside. Once a worm uses IM and P2P to get past perimeter defences, it may have great success spreading internally by file shares, email, IM or other techniques.
2. Buddy lists – even more than in email, people seem to trust IMs from people they know.
1. Dude – it's *AOL* and *Microsoft* – there's nothing evil about attacking them!

MANAGING IM AND P2P – A CHALLENGE

Effectively blocking IM and P2P traffic is difficult. The first response of most IT organizations concerned about IM and P2P usage is to seek to block the ports used by IM. A quick look at some of the IM clients will show that they default to ports such as 1863, 4662, 4672, 5050, 5190 and 6881–6999. Blocking these ports at the firewall will stop much of the IM and P2P traffic – but not all. The port usage of many IM and P2P applications is dynamic, i.e. they use any open TCP/UDP port to communicate with another P2P host, and from there connect to many other hosts. As a result, filtering these programs with an access list is ineffective.

In addition, some real-time communications applications, such as *Yahoo! Messenger* and *Skype* will automatically revert to HTTP tunnelling if their default ports are unavailable.

Because server IP addresses and URLs used by IM and P2P products change on a regular basis, attempts to block IM and P2P by IP and URL filtering is similarly ineffective unless you have the resources to track continuously the addresses being used and update your filters.

Adding to the challenge of managing IM and P2P is the rapid technological churn in this arena. New P2P protocols and applications are turning all the time. The relative popularity of the various networks waxes and wanes so quickly that just keeping up with the latest is a full-time job. *BitTorrent* for example, which was first released in 2002, now may account for the use of more Internet bandwidth than any other protocol on the planet – including HTTP. *Skype*, which barely existed a year ago, will soon have had more than 40 million downloads, *Voiceglo* has announced a 'converged' instant messaging client with voice capability to challenge *Skype*, *Microsoft* has announced new links between its *MSN Messenger*, *HotMail* and its new *MSN Spaces* blogging tool.

The unfortunate truth is that instant messaging and peer-to-peer networks represent yet another area of security where specialized expertise and focus is urgently required.

COMPARATIVE REVIEW

WINDOWS NT

Matt Ham

I seem to remember having been aware in the last comparative review on this platform (see *VB*, February 2004, p.12) of a sense of impending doom that accompanied *Windows NT 4* – a sense of doom which is rather more pronounced a year later. With *Microsoft* having decided to remove support for *NT 4*, it must have been tempting for other developers to do much the same, if only to save on back-compatibility testing resources. However, it seems that *Trend Micro* is the only vendor that has opted to remove support at this stage – which explains the absence of a *Trend* product in the review.

I would not be willing to suggest that many other companies will follow suit. Having seen some veritably antique hardware and software in use, even in supposedly high-end research environments, I suspect there will continue to be a market for *NT* products for years to come. With large customers being able to blackmail legacy support from the vendors, it can be hard simply to terminate support for an otherwise unattractive platform.

Of the products submitted, *Hauri*'s proved to be the most beset with problems. An initial version had severe issues with resources, rendering it incomprehensible to mortal man. A replacement version proved to cause sufficient instability on access that testing was all but impossible. Therefore the product was left alone after these tribulations. A host of new product arrivals, however, pushed the number of contenders in this review to 28 – an all-time record for *VB*'s comparatives.

THE TEST SETS

The test sets were aligned to the Real-Time WildList from October 2004, the newer WildList arriving, as luck would have it, a day after the deadline for product submissions.

The additions to the set were, as is becoming rather a predictable occurrence, all immutable worms and far larger in number than the more interesting specimens that no longer appear in the wild. If ever a file infector comes into the wild again, I will at least find the process of replicating the test sets a degree more interesting.

AhnLab V3 VirusBlock 6.0.0.312

ItW Overall	99.75%	**Macro**	98.97%
ItW Overall (o/a)	100.00%	**Standard**	96.18%
ItW File	99.75%	**Polymorphic**	63.81%

Reappearing after a brief absence from *VB*'s comparative reviews, *V3* came very close to achieving a VB 100% award this month, missing only one sample of W32/Bagle.BB on demand. Elsewhere, *V3*'s polymorphic performance is still somewhat weak, though detection in the other sets has improved since the last tests.

Alwil avast! 4.5.555

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

In a repeat performance of other recent tests of the product, the *avast!* review started with the on-access service failing due to the blank password on the test platform. One changed password later, however, all problems had vanished and avast! earned itself a VB 100%.

ArcaBit ArcaVir 2005

ItW Overall	99.64%	**Macro**	98.52%
ItW Overall (o/a)	99.71%	**Standard**	97.87%
ItW File	99.64%	**Polymorphic**	85.48%

ArcaVir has appeared in *Virus Bulletin*'s tests before, albeit under the name of *MKS*. The product has been rebadged, retuned and re-released, with the intention of marketing it to a more international audience. A handful of misses in the ItW set and a false positive blemished an otherwise impressive debut. Since the missed files were, in many cases, missed as a result of extension issues, the result should be improved upon in forthcoming reviews.

Less impressive, however, was the requirement to find and install MFC42.DLL manually before the program would operate.

Authentium Command 4.92.7

ItW Overall	100.00%	**Macro**	100.00%
ItW Overall (o/a)	100.00%	**Standard**	99.72%
ItW File	100.00%	**Polymorphic**	99.95%

Unlike the previous product, *Command* is a long-standing and familiar name. Seeming to work just as well on *NT* as on the rather newer *XP*, another VB 100% award goes to *Authentium*.

On-access tests	ItW File		ItW Boot		ItW Overall	Macro		Polymorphic		Standard	
	Number missed	%	Number missed	%	%	Number missed	%	Number missed	%	Number missed	%
AhnLab V3 VirusBlock	0	100.00%	0	100.00%	100.00%	47	98.97%	5549	63.81%	54	96.18%
Alwil avast!	0	100.00%	0	100.00%	100.00%	18	99.56%	112	93.58%	17	99.18%
ArcaBit ArcaVir 2005	2	99.70%	0	100.00%	99.71%	33	99.47%	1402	85.48%	33	97.94%
Authentium Command	0	100.00%	0	100.00%	100.00%	0	100.00%	1	99.95%	5	99.58%
Avira Avira	0	100.00%	0	100.00%	100.00%	0	100.00%	4	99.63%	6	99.91%
BLC Win Cleaner	0	100.00%	0	100.00%	100.00%	86	97.96%	1477	91.03%	473	72.47%
CA eTrust Antivirus (InoculateIT)	0	100.00%	0	100.00%	100.00%	4	99.90%	1	99.89%	4	99.51%
CA eTrust Antivirus (Vet)	0	100.00%	0	100.00%	100.00%	12	99.82%	1	99.92%	5	99.60%
CA Vet Anti-Virus	0	100.00%	0	100.00%	100.00%	0	100.00%	2	99.87%	5	99.60%
CAT Quick Heal	0	100.00%	0	100.00%	100.00%	86	97.96%	1477	91.03%	473	72.47%
Doctor Web Dr.Web	0	100.00%	0	100.00%	100.00%	0	100.00%	0	100.00%	3	99.69%
Eset NOD32	0	100.00%	0	100.00%	100.00%	0	100.00%	0	100.00%	0	100.00%
Fortinet FortiClient	0	100.00%	0	100.00%	100.00%	200	95.85%	5656	61.54%	63	97.90%
FRISK F-Prot Antivirus	1	99.96%	0	100.00%	99.96%	0	100.00%	6	99.97%	10	99.19%
F-Secure Anti-Virus	3	99.24%	0	100.00%	99.24%	0	100.00%	0	100.00%	3	99.85%
GDATA AntiVirusKit	0	100.00%	0	100.00%	100.00%	0	100.00%	0	100.00%	0	100.00%
Grisoft AVG	0	100.00%	0	100.00%	100.00%	19	99.53%	757	83.64%	33	98.35%
H+BEDV AntiVir	0	100.00%	0	100.00%	100.00%	0	100.00%	7	99.59%	5	99.92%
Kaspersky KAV	0	100.00%	0	100.00%	100.00%	0	100.00%	0	100.00%	2	99.88%
McAfee VirusScan	1	99.75%	0	100.00%	99.75%	0	100.00%	0	100.00%	3	99.79%
MicroWorld eScan	0	100.00%	0	100.00%	100.00%	0	100.00%	0	100.00%	0	100.00%
Norman Virus Control	0	100.00%	0	100.00%	100.00%	2	99.95%	181	91.03%	12	99.45%
NWI Virus Chaser	0	100.00%	0	100.00%	100.00%	0	100.00%	0	100.00%	3	99.69%
SOFTWIN BitDefender	0	100.00%	0	100.00%	100.00%	35	99.17%	6	99.73%	14	99.05%
Sophos Anti-Virus	0	100.00%	0	100.00%	100.00%	8	99.80%	0	100.00%	15	99.30%
SR Resolution Antivirus	17	97.79%	3	0.00%	97.05%	20	99.58%	1014	88.96%	28	99.17%
Symantec SAV	0	100.00%	0	100.00%	100.00%	0	100.00%	0	100.00%	0	100.00%
UNA UNA	414	16.79%	3	0.00%	16.67%	247	94.23%	15140	0.00%	22	97.43%
VirusBuster VirusBuster	0	100.00%	0	100.00%	100.00%	0	100.00%	96	92.79%	16	99.17%

Avira Avira 1.00.00.61

ItW Overall	100.00%	**Macro**	100.00%
ItW Overall (o/a)	100.00%	**Standard**	99.78%
ItW File	100.00%	**Polymorphic**	99.67%

Another product arriving in hopes of a new international audience, *Avira* will be better recognised by many readers as a prettier version of *H+BEDV*'s *AntiVir*.

With a few previous reviews from which to learn the ropes, the results achieved by this ostensibly new product were good indeed – and were certainly sufficient to be rewarded with a VB 100%.

BLC Win Cleaner 7.03

ItW Overall	100.00%	Macro	98.00%
ItW Overall (o/a)	100.00%	Standard	96.39%
ItW File	100.00%	Polymorphic	92.85%

Continuing in the same vein, *Win Cleaner* is, as assiduous readers will remember, a rebadged version of *CAT*'s *Quick Heal*. The physical resemblance here is very great indeed, as is the detection quality. Another VB 100% results.

CA eTrust Antivirus (I) 7.1.192

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

This is the non-default version of *eTrust*, using the *InoculateIT* engine. After a slight hiccough in the last test, the product returned to put in a good performance on this occasion. Since this is the non-default version of *eTrust*, and its inclusion in the tests is for reasons of comparison with its *Vet*-engined counterpart, no VB 100% is awarded.

CA eTrust Antivirus (V) 7.1.192

ItW Overall	100.00%	Macro	99.82%
ItW Overall (o/a)	100.00%	Standard	99.72%
ItW File	100.00%	Polymorphic	99.87%

Not to be outdone by its different-engined sibling, the *Vet*-powered version of *eTrust* also put in a good performance and this, the default version of the product, is awarded a VB 100%. Sadly, the two products share what is, in my opinion, the most feeble and useless logging system ever devised by an otherwise reliable developer.

CA Vet Anti-Virus 10.6.4.0.9

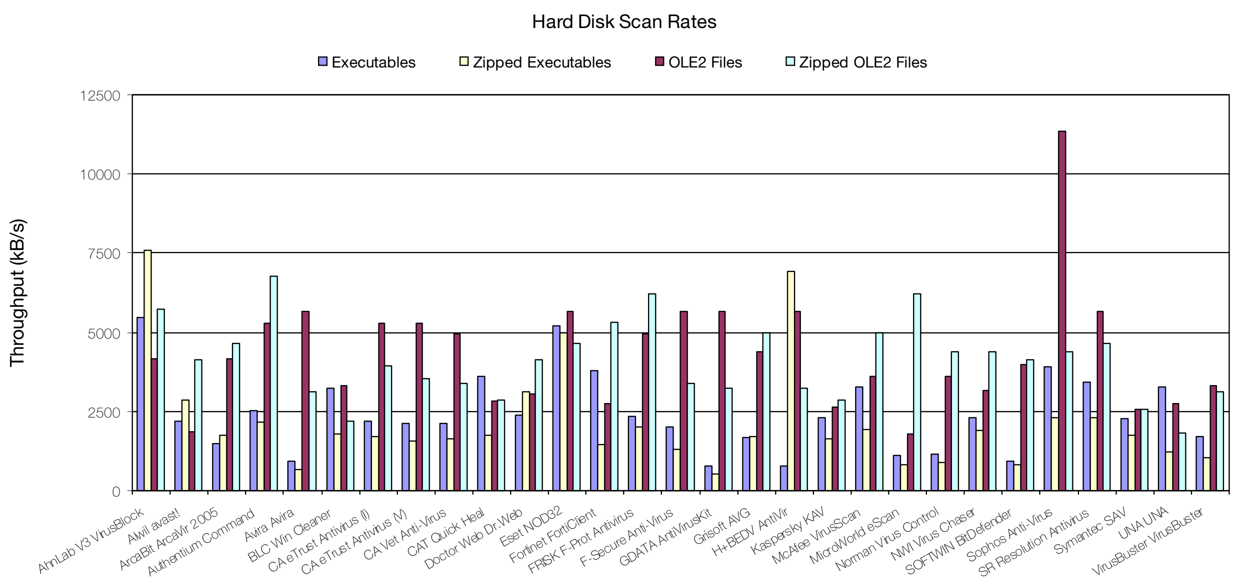
ItW Overall	100.00%	Macro	100.00%
ItW Overall (o/a)	100.00%	Standard	99.72%
ItW File	100.00%	Polymorphic	99.87%

Using the same engine as the previous contender, the all-*Vet* product rejoices in a rather better logging system. Happily, in gaining this advantage it has lost no efficiency, and it too receives a VB 100% award.

CAT Quick Heal 7.03

ItW Overall	100.00%	Macro	98.00%
ItW Overall (o/a)	100.00%	Standard	96.39%
ItW File	100.00%	Polymorphic	92.85%

Since its 'offspring' has already received a VB 100% award in this test, it should come as little surprise that *CAT* does too. The interface remains somewhat sparse, but certainly contains all the functionality required.



Hard Disk Scan Rate	Executables			OLE Files			Zipped Executables		Zipped OLE Files	
	Time (s)	Throughput (KB/s)	FPs [susp]	Time(s)	Throughput (KB/s)	FPs [susp]	Time (s)	Throughput (KB/s)	Time(s)	Throughput (KB/s)
AhnLab V3 VirusBlock	100	5469.3		19	4175.5		21	7591.3	13	5739.0
Alwil avast!	251	2179.0		43	1845.0		56	2846.7	18	4144.9
ArcaBit ArcaVir 2005	365	1498.4	1	19	4175.5		92	1732.8	16	4663.0
Authentium Command	215	2543.9		15	5288.9		74	2154.3	11	6782.5
Avira Avira	579	944.6		14	5666.7		232	687.1	24	3108.6
BLC Win Cleaner	169	3236.3		24	3305.6		89	1791.2	34	2194.3
CA eTrust Antivirus (InoculateIT)	249	2196.5		15	5288.9		93	1714.2	19	3926.7
CA eTrust Antivirus (Vet)	256	2136.5		15	5288.9		101	1578.4	21	3552.7
CA Vet Anti-Virus	257	2128.1		16	4958.4		98	1626.7	22	3391.2
CAT Quick Heal	152	3598.2		28	2833.3		91	1751.8	26	2869.5
Doctor Web Dr.Web	228	2398.8		26	3051.3		51	3125.8	18	4144.9
Eset NOD32	105	5208.9		14	5666.7		32	4981.8	16	4663.0
Fortinet FortiClient	144	3798.1		29	2735.6		110	1449.2	14	5329.1
FRISK F-Prot Antivirus	232	2357.5		16	4958.4		80	1992.7	12	6217.3
F-Secure Anti-Virus	270	2025.7		14	5666.7		121	1317.5	22	3391.2
GDATA AntiVirusKit	695	787.0		14	5666.7		300	531.4	23	3243.8
Grisoft AVG	329	1662.4		18	4407.4		93	1714.2	15	4973.8
H+BEDV AntiVir	705	775.8		14	5666.7		23	6931.2	23	3243.8
Kaspersky KAV	239	2288.4		30	2644.5		97	1643.5	26	2869.5
McAfee VirusScan	167	3275.0		22	3606.1		83	1920.7	15	4973.8
MicroWorld eScan	497	1100.5	1	44	1803.0		191	834.6	12	6217.3
Norman Virus Control	482	1134.7		22	3606.1		180	885.6	17	4388.7
NWI Virus Chaser	237	2307.7		25	3173.4		84	1897.8	17	4388.7
SOFTWIN BitDefender	591	925.4		20	3966.7		198	805.1	18	4144.9
Sophos Anti-Virus	140	3906.7		7	11333.4		69	2310.4	17	4388.7
SR Resolution Antivirus	160	3418.3		14	5666.7		69	2310.4	16	4663.0
Symantec SAV	240	2278.9		31	2559.2		91	1751.8	29	2572.7
UNA UNA	167	3275.0		29	2735.6		131	1216.9	41	1819.7
VirusBuster VirusBuster	319	1714.5	[1]	24	3305.6		154	1035.2	24	3108.6

Doctor Web Dr.Web 4.32b

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

Having recently changed ownership, there was potential for changes in the *Dr.Web* product, for better or for worse. However, since the new owner is the active developer of the product, it is not surprising that there have not been sweeping changes. A solid performance earns the product a VB 100% award.

Eset NOD32 1.956

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

On-demand tests	ItW File		ItW Boot		ItW Overall	Macro		Polymorphic		Standard	
	Number missed	%	Number missed	%	%	Number missed	%	Number missed	%	Number missed	%
AhnLab V3 VirusBlock	1	99.75%	0	100.00%	99.75%	47	98.97%	5549	63.81%	54	96.18%
Alwil avast!	0	100.00%	0	100.00%	100.00%	18	99.56%	112	93.58%	15	99.36%
ArcaBit ArcaVir 2005	5	99.64%	0	100.00%	99.64%	92	98.52%	1402	85.48%	32	97.87%
Authentium Command	0	100.00%	0	100.00%	100.00%	0	100.00%	1	99.95%	2	99.72%
Avira Avira	0	100.00%	0	100.00%	100.00%	0	100.00%	1	99.67%	11	99.78%
BLC Win Cleaner	0	100.00%	0	100.00%	100.00%	82	98.00%	1086	92.85%	101	96.39%
CA eTrust Antivirus (InoculateIT)	0	100.00%	0	100.00%	100.00%	4	99.90%	1	99.89%	0	100.00%
CA eTrust Antivirus (Vet)	0	100.00%	0	100.00%	100.00%	12	99.82%	2	99.87%	3	99.72%
CA Vet Anti-Virus	0	100.00%	0	100.00%	100.00%	0	100.00%	2	99.87%	3	99.72%
CAT Quick Heal	0	100.00%	0	100.00%	100.00%	82	98.00%	1086	92.85%	101	96.39%
Doctor Web Dr.Web	0	100.00%	0	100.00%	100.00%	0	100.00%	0	100.00%	0	100.00%
Eset NOD32	0	100.00%	0	100.00%	100.00%	0	100.00%	0	100.00%	0	100.00%
Fortinet FortiClient	0	100.00%	0	100.00%	100.00%	157	96.65%	5648	61.57%	87	97.55%
FRISK F-Prot Antivirus	0	100.00%	0	100.00%	100.00%	0	100.00%	0	100.00%	2	99.72%
F-Secure Anti-Virus	0	100.00%	0	100.00%	100.00%	0	100.00%	0	100.00%	1	99.98%
GDATA AntiVirusKit	0	100.00%	0	100.00%	100.00%	0	100.00%	0	100.00%	0	100.00%
Grisoft AVG	0	100.00%	0	100.00%	100.00%	20	99.51%	257	85.97%	27	98.56%
H+BEDV AntiVir	0	100.00%	0	100.00%	100.00%	0	100.00%	1	99.67%	7	99.90%
Kaspersky KAV	0	100.00%	0	100.00%	100.00%	0	100.00%	0	100.00%	0	100.00%
McAfee VirusScan	1	99.75%	0	100.00%	99.75%	0	100.00%	0	100.00%	3	99.79%
MicroWorld eScan	0	100.00%	0	100.00%	100.00%	0	100.00%	0	100.00%	0	100.00%
Norman Virus Control	0	100.00%	0	100.00%	100.00%	2	99.95%	180	91.24%	6	99.66%
NWI Virus Chaser	0	100.00%	0	100.00%	100.00%	0	100.00%	0	100.00%	1	99.82%
SOFTWIN BitDefender	0	100.00%	0	100.00%	100.00%	34	99.12%	6	99.73%	22	99.23%
Sophos Anti-Virus	0	100.00%	0	100.00%	100.00%	8	99.80%	0	100.00%	15	99.30%
SR Resolution Antivirus	0	100.00%	3	0.00%	99.24%	2	99.93%	1015	88.95%	14	99.63%
Symantec SAV	0	100.00%	0	100.00%	100.00%	0	100.00%	0	100.00%	0	100.00%
UNA UNA	32	93.05%	3	0.00%	92.35%	1914	54.54%	14266	20.19%	517	75.89%
VirusBuster VirusBuster	0	100.00%	0	100.00%	100.00%	0	100.00%	98	92.78%	13	99.31%

With its usual admirable performance, *NOD32* once again leaves little room for comment and achieves its latest VB 100% award with predictable ease.

Perhaps a revamp of the product's GUI is in order, complete with Easter egg functionality for those reviewers at a loss for sensible comment.

Fortinet FortiClient 1.2.1134

ItW Overall	100.00%	**Macro**	96.65%
ItW Overall (o/a)	100.00%	**Standard**	97.55%
ItW File	100.00%	**Polymorphic**	61.57%

Although still often tricked by polymorphic samples, *FortiClient* continues to improve its performance in other areas. Now that VB 100% status looks to be achievable by the product on a regular basis, it seems likely that the developers will be pushed towards working on the matter of detecting those polymorphics.

FRISK F-Prot Antivirus 3.16 a

ItW Overall	100.00%	**Macro**	100.00%
ItW Overall (o/a)	99.96%	**Standard**	99.72%
ItW File	100.00%	**Polymorphic**	100.00%

Matters with *F-Prot* were much the same as ever, even down to the miss of W32/Nimda in its .EML form, due to the decision not to scan such files on access. This is clearly a decision that has been based upon speed of scanning being regarded as an important feature, since large email files will by their nature be most scanners' worst nightmare.

F-Secure Anti-Virus 5.43

ItW Overall	100.00%	**Macro**	100.00%
ItW Overall (o/a)	99.24%	**Standard**	99.98%
ItW File	100.00%	**Polymorphic**	100.00%

F-Secure Anti-Virus seemed something of a changed product this month, in comparison with its performance on other platforms. On several occasions the test machine blue-screened on boot, though on the occasions when a blue screen did not occur, the product was working at close to its usual level of efficiency. There were, however, misses on some of the newer worms in the test set, which denied *FSAV* a VB 100%.

GDATA AntiVirusKit 14.1.

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

It came as something of a surprise when *AVK* delivered two blue screens while on-access scanning was in progress. The problem seemed to be dissipated when logging was disabled, though admittedly the sample set was not large enough to make definite pronouncements.

Despite this momentary excitement, detection was as good as usual and a VB 100% was the due reward for *GDATA*.

Grisoft AVG 7.0.290

ItW Overall	100.00%	**Macro**	99.51%
ItW Overall (o/a)	100.00%	**Standard**	98.56%
ItW File	100.00%	**Polymorphic**	85.97%

With *AVG* version 7 now firmly in place, all momentary problems from interface changes are well and truly over. The result is a product which is simple to review and, rather more happily for the developers, gains another VB 100%.

H+BEDV AntiVir 6.29.00.03

ItW Overall	100.00%	**Macro**	100.00%
ItW Overall (o/a)	100.00%	**Standard**	99.90%
ItW File	100.00%	**Polymorphic**	99.67%

Much like its progeny, *Avira*, *H+BEDV* performed amply well enough to receive a VB 100% award. The future of *H+BEDV* is something of a mystery though – will it remain as a free product or will it be subsumed by *Avira*? Only time will tell.

Kaspersky KAV 5.0.277

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

Kaspersky seems, in this latest version, to have removed the emission of annoying noises by default upon virus detection. Deprived of this perennial complaint, I shall be forced to revert to commenting on more technical matters. The product behaved impeccably and gained a VB 100% award.

McAfee VirusScan 8.0.0 4415

ItW Overall	99.75%	**Macro**	100.00%
ItW Overall (o/a)	100.00%	**Standard**	99.79%
ItW File	99.75%	**Polymorphic**	100.00%

Usually a stalwart on the matter of detections, it came as something of a surprise when *VirusScan* failed to detect one of the newer worms in the test set. With the multitudes of such creations produced each day, such a miss is not surprising on occasion, though the developers at *McAfee* will not, I suspect, take a great deal of comfort from this fact.

MicroWorld eScan 1.2.7

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

Since *eScan* is based on *AVK*, it was of interest to see whether the unusual behaviour displayed by *AVK* in this test would be repeated. There were no blue screens here, however – making *AVK*'s problem even more perplexing. As a problem of its own, *eScan* produced a false positive in the clean sets. This effectively denied the product a VB 100%

Norman Virus Control 5.7.0

ItW Overall	100.00%	**Macro**	99.95%
ItW Overall (o/a)	100.00%	**Standard**	99.66%
ItW File	100.00%	**Polymorphic**	91.24%

As with several other products in the test, *Norman*'s sole weakness lies in the polymorphic test set. This seems rather strange, since *Norman* does have available some much-vaunted heuristic techniques.

Despite this weakness, however, good detection rates and a lack of false positives combined earn *NVC* a VB 100% on this occasion.

NWI Virus Chaser 5.0

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

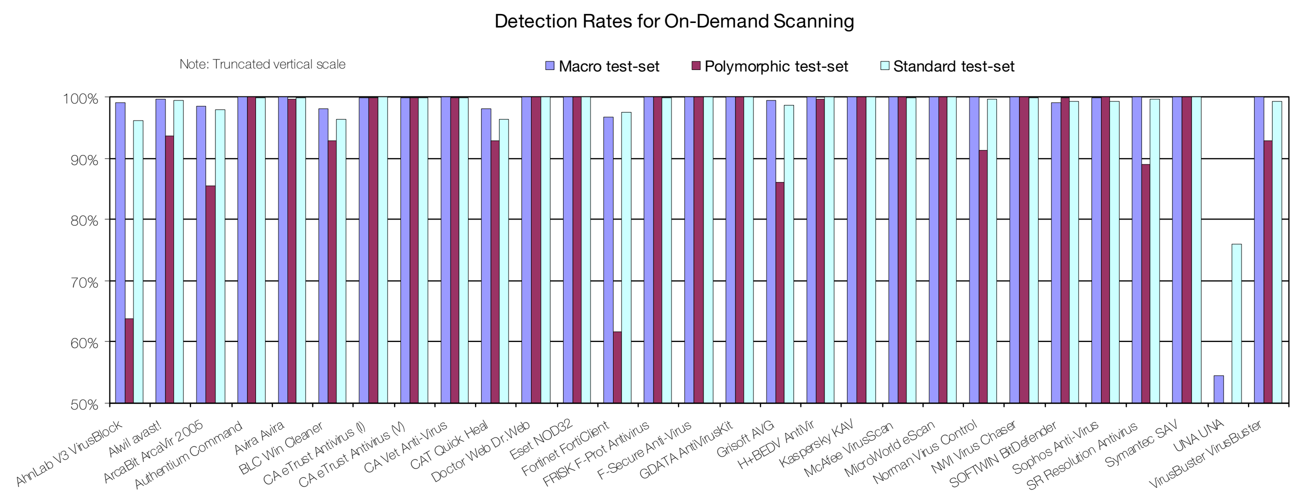
As a rebadged version of *Dr.Web*, the excellent detection rates of this product come as no surprise.

The overbearing rigidity of the user interface, however, makes it unpleasant to use. One good feature, though, is that, unlike *Dr.Web*'s interface, it is possible to change on-access settings without rebooting the machine. It is possible, however, that this is simply due to the more limited changes available.

SOFTWIN BitDefender 8.0.137

ItW Overall	100.00%	**Macro**	99.12%
ItW Overall (o/a)	100.00%	**Standard**	99.23%
ItW File	100.00%	**Polymorphic**	99.73%

Although currently rather annoyed by the fact that parts of their program have been declared spyware by *Microsoft*, the developers at *SOFTWIN* should at least be consoled by the fact that a solid performance by the *BitDefender* product this month has earned them a new VB 100% award to place upon their virtual mantelpiece.



Sophos Anti-Virus 3.88.0

ItW Overall	100.00%	Macro	99.80%
ItW Overall (o/a)	100.00%	Standard	99.30%
ItW File	100.00%	Polymorphic	100.00%

Sophos Anti-Virus produced all but identical results to those it has produced in the previous tests. A VB 100% was thus awarded. After a burst of welcome improvements to the product in the middle part of 2004 it remains to be seen whether the improvements continue in 2005.

SR Resolution Antivirus

ItW Overall	99.24%	Macro	99.93%
ItW Overall (o/a)	97.05%	Standard	99.63%
ItW File	100.00%	Polymorphic	88.95%

This rebadged *Panda* product fared reasonably well in most aspects. Floppy scanning, however, was either ineffective on access, or ineffective and caused the program to shut down while producing large error warnings. Non-floppy scanning was better, though with extensionless and PIF files remaining unscanned on access, there is room for easy improvement here.

Symantec SAV 9.0.0.338

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

Having harped on somewhat about the slowness of scanning in previous *Symantec* tests I embarked on some limited experiments on this occasion. By disabling the screen updates for scanning progress and all requirements to process files after scanning, the delays during scanning were banished completely. It seems, therefore, that any slowness is at least partially GUI-related rather than an issue with the engine itself. These matters enlivened the otherwise predictable arrival of another VB 100% for *Symantec*.

UNA UNA 1.83

ItW Overall	92.35%	Macro	54.54%
ItW Overall (o/a)	16.67%	Standard	75.89%
ItW File	93.05%	Polymorphic	20.19%

On this occasion *UNA* certainly wins the prize for the most disparate set of results on access and on demand. Although the statistics show great differences, the underlying details were even more perplexing. The number of potential variables here makes hazarding an explanation rather futile.

VirusBuster VirusBuster 4.7.22

ItW Overall	100.00%	Macro	100.00%
ItW Overall (o/a)	100.00%	Standard	99.31%
ItW File	100.00%	Polymorphic	92.78%

With a suspicious file in the clean sets, *VirusBuster* teetered perilously close to missing out on a VB 100%. The file was not declared to be viral, however and so full detection in the ItW sets was ample to earn *VirusBuster* a further VB 100%.

CONCLUSION

The results from this test brought two totally unrelated thoughts to my mind. The first is the nature of the test sets. The increase in very similar worm additions to the WildList, identified by checksums rather than names, is the result of a vast flood of similar files entering into circulation. Detecting these files is not usually an issue – a good archive-handling engine is a great help, admittedly, but by and large they are not a massive challenge. Gathering these files is, however, much more of an issue – the burden upon the developers is ever more veering towards a logistic problem rather than a detection challenge.

The second thought relates to the nature of older platforms. On *XP* I very rarely see any problems with anti-virus programs. The reasons for this are probably split between extra development effort being expended on this platform and the rather more robust *XP* structure. With *NT,* however, there were several occasions where products simply curled up and died in a sea of blue. Overcoming such problems on an aged platform, while not breaking newer platforms, is a challenge that will give coders a few sleepless nights.

Technical details:

Test environment: Three 1.6 GHz Intel Pentium 4 workstations with 512 MB RAM, 20 GB dual hard disks, DVD/CD-ROM and 3.5-inch floppy, all running *Windows NT 4 Workstation Service Pack 6*.

Virus test sets: Complete listings of the test sets used can be found at http://www.virusbtn.com/Comparatives/WinNT/2005/test_sets.html.

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

END NOTES & NEWS

SecureLondon 2005 takes place 10 February 2005 in London, UK. For more information, and to register, visit https://www.isc2.org/.

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

Websec 2005: i-Security World Conference takes place 15–17 March 2005 in London, UK. The conference features three tracks: security policy, risk & governance; IT infrastructure security; and enterprise application security. See http://www.mistieurope.com/.

The E-crime and Computer Evidence conference ECCE 2005 takes place in Monaco from 29–30 March 2005. ECCE 2005 will consider aspects of digital evidence in all types of criminal activity, including timelines, methods of evidence deposition, use of computers for court presentation, system vulnerabilities, crime prevention etc. For more details see http://www.ecce-conference.com/.

Black Hat Europe takes place in Amsterdam, The Netherlands, from 29 March to 1 April 2005. Black Hat Europe Training runs from 29 to 30 March, with the Black Hat Europe Briefings following, from 31 March until 1 April.

Black Hat Asia takes place 5–8 April 2005 in Singapore. The Briefings take place 5–6 April, with the training on 7–8 April. For details and registration see http://www.blackhat.com/.

The first Information Security Practice and Experience Conference (ISPEC 2005) will be held 11–14 April 2005 in Singapore. ISPEC is intended to bring together researchers and practitioners to provide a confluence of new information security technologies, their applications and their integration with IT systems in various vertical sectors. For more information see http://ispec2005.i2r.a-star.edu.sg/.

Infosecurity Europe 2005 takes place 26–28 April 2005 in London, UK. There will be more than 250 exhibitors and the organisers expect over 10,000 visitors. See http://www.infosec.co.uk/.

The 14th EICAR conference will take place from 30 April to 3 May 2005 in Saint Julians, Malta. A call for poster submissions remains open until 18 February 2005. See http://conference.eicar.org/.

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

The third International Workshop on Security in Information Systems, WOSIS-2005, takes place 24–25 May 2005 in Miami, USA. For full details see http://www.iceis.org/.

The 3rd annual BCS IT Security Conference takes place on 7 June 2005 in Birmingham, UK. The conference focuses on identity theft, hacking, cyber-terrorism, network forensics, secure web services, encryption and related topics. See http://www.bcsinfosec.com/.

NetSec 2005 will be held 13–15 June 2005 in Scottsdale AZ, USA. The program covers a broad array of topics, including awareness, privacy, policies, wireless security, VPNs, remote access, Internet security and more. See http://www.gocsi.com/events/netsec.jhtml.

A SRUTI 2005 workshop entitled 'Steps to Reducing Unwanted Traffic on the Internet' takes place 7–8 July 2005 in Cambridge, MA, USA. The Usenix-sponsored workshop aims to bring academic and industrial research communities together with those who face the problems at the operational level. For more information see http://www.research.att.com/~bala/sruti/.

Black Hat USA takes place 23–28 July 2005 in Las Vegas, NV, USA. A call for papers opens on 15 February 2005. For details see http://www.blackhat.com/.

The 14th USENIX Security Symposium will be held 1–5 August 2005 in Baltimore, MD, USA. For more information see http://www.usenix.org/.

The 15th Virus Bulletin International Conference, VB2005, will take place 5–7 October 2005 in Dublin, Ireland. For conference registration, sponsorship and exhibition information and details of how to submit a paper see http://www.virusbtn.com/.

SUBSCRIPTION RATES

Subscription price for 1 year (12 issues) including first-class/airmail delivery: £195 (US$358)

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

TSUNAMI UNLOCKS FLOODGATES FOR OPPORTUNISTS

As most of the world was still reeling from the news and pictures of devastation in Asia following the 26 December tsunami, the FBI was forced to issue an alert last month, warning those wishing to donate to tsunami relief funds that they may be targeted by Trojan exploits and 419 scams.

The FBI reported that bogus websites had been set up masquerading as legitimate relief organizations requesting donations – at least one of which, it stated, contained an embedded Trojan exploit. In other scams, those who had made appeals for information about friends and relatives still missing following the tsunami were targeted by unsolicited emails that offered to locate loved ones – for a fee. And, in the UK, a 40-year-old man was jailed for six months after being found guilty of sending hoax emails to relatives and friends of the missing, stating that the UK government 'regretted to inform the victim that the missing person they were inquiring about was confirmed dead'. The man claimed he had suffered a 'moment of madness' when he concocted the messages.

Of course, '419ers' never miss a trick, and the Internet has swarmed with an influx of messages requesting that money be deposited in overseas banks to support the tsunami relief effort or asking for personal or financial information in an effort to retrieve inheritance funds tied up in relation to the tsunami disaster. Indeed, many a security-savvy sysadmin may have felt ostracised by colleagues horrified that they had taken the decision to block these and other apparent 'desperate pleas for help'.

In the US the FBI arrested a man last month for sending around 800,000 hoax tsunami fund-raising messages. The FBI tracked down Matthew Schmieder, from Pittsburgh, with the help of UK anti-spam operation *Spamhaus*. Unlike most regular spammers, Schmieder had made little attempt to cover his tracks. *Spamhaus*'s Steve Linford said: 'He had very little in place by way of defences and … we were able to very quickly track him down … He lived right around the corner from the FBI offices.'

In the UK, an attempt to hack into the website of the Disasters and Emergency Committee (DEC), which was set up after the tsunami, is currently under investigation. A 28-year-old man has been arrested and is being questioned.

The (hopefully) final and depressingly inevitable piece to the tsunami jigsaw in terms of IT security knock-on effects has been the discovery of mass-mailing worm W32/Zar@mm, which poses as a plea for donations to help with the tsunami disaster and the VBS/Geven worm, which claimed that the tsunami was God's revenge on 'people who did bad on earth'.

More encouragingly, however, *VB* is pleased also to be able to report the better side of human nature: a number of authors of recent *VB* articles have requested that their honorarium payments be donated to tsunami relief funds.

EVENTS

The Messaging Anti-Abuse Working Group (MAAWG) will hold its third general meeting 1–3 March 2005 in San Diego, CA, USA. The meeting is open to non-members and members of MAAWG. See http://www.maawg.org/.

CEAS 2005, the Second Conference on Email and Anti-Spam, will be held 21–22 July 2005 at Stanford University, CA, USA. The conference committee is currently seeking submissions for papers. For details see http://www.ceas.cc/.

INBOX IT is planned for early June 2005 in the San Francisco Bay area, CA, USA. More information will be available in due course from http://www.inboxevents.com/.

TREC 2005, the Text Retrieval Conference, will be held 15–18 November 2005 at NIST in Gaithersburg, MD, USA. A new track on spam aims to provide a standard evaluation of current and proposed spam filtering approaches. For more information see http://trec.nist.gov/.

FEATURE

COMBINING MULTIPLE CLASSIFIERS

Richard Segal
IBM Research, USA

There are many techniques for filtering spam, each with its own benefits and limitations. For instance, real-time blacklists (RBL) are effective only at blocking spam attacks that originate from a small set of IP addresses. Techniques that compare the signature of incoming mail to a database of signatures of known spam can block spam that has previously been detected by other users, but perform poorly on new attacks. Naïve-bayesian filters are highly effective in most instances, but can produce false positives and are open to gaming by determined spammers.

However, a spam filter that combines multiple classifiers has the potential to take advantage of each classifier's strengths while working around each classifier's weaknesses. The end result is a classifier that can catch more spam with fewer false positives than any of its constituent parts.

The value of combining multiple classifiers goes beyond predictive ability. Multiple-classifier systems are more difficult to attack because it is less likely that any single adjustment a spammer makes to an email campaign can fool all of the classifiers that are used in a multi-classifier system. If some of the classifiers are not fooled, the spammer's campaign will still be blocked.

COMBINATION METHODS

Assume that each classifier rates the 'spamminess' of incoming messages by returning a score from 0 to 1000. A score of 0 indicates that the message is almost certainly good, a score of 1000 indicates that the message is almost certainly spam. The output of most classifiers can be scaled to fit this range as needed. The score is compared against a user-defined threshold and all mail with a score above this threshold is blocked. Allowing the user to set a filtering threshold enables the user to choose what trade-off they prefer between the amount of spam caught and the number of false positives.

The scores of several different classifiers can be combined to compute a single score. There are several methods for combining scores. One option is to return the minimum score output by any of the classifiers. Using this method a message will be labelled as spam only if all the classifiers return a score higher than the user-provided threshold. That is, all the classifiers agree that the message is spam. The minimum score aggregator produces a very low false positive rate, since a good message can be misclassified only if all the algorithms label the message incorrectly as spam. On the other hand, its spam detection rate can be no better than the least effective classifier. Minimum score aggregators are not resilient to attack because a spammer needs only to attack one classifier successfully to get their mail through.

A better approach is to combine the scores of each classifier using a weighted average. This method is more resilient to attack since the opinion of all of the classifiers contributes to every prediction. If we denote the score of a classifier on example x as $S_i(x)$ and the score of the combined classifier as $S(x)$, the score assigned to the combination of N classifiers is:

$$S(x) = \sum_{i=1..N} w_i * S_i(x)$$

The weights (w_i) should be chosen to reflect the relative strengths of each classifier. If the relative strengths are unknown or each classifier has comparable performance, equal weights can be used by setting $w_i = 1$ for all classifiers.

Equal weights is often the best approach. While small differences may exist in the strengths of each classifier, these differences are often not enough to have a substantial effect upon the accuracy of the combined classifier. Equal weight aggregators are also more resilient to attack than non-equal weight aggregators because no classifier has sufficient weight such that a successful attack against that

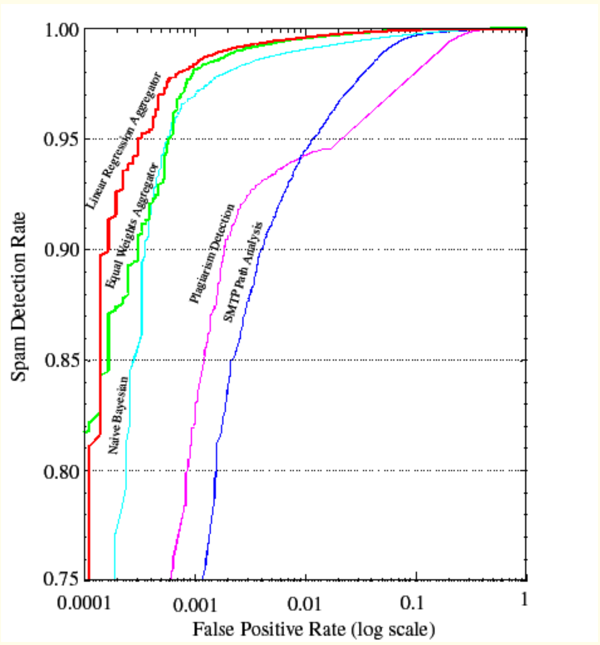


Figure 1. ROC curve for different spam classification algorithms.

classifier can substantially impact performance. While better performance can be achieved by a more precise selection of weights, that performance comes at the cost of greater susceptibility to attack.

Figure 1 shows a ROC (Receiver Operating Characteristics) curve for three classification algorithms and an equal weights aggregator that combines the algorithms shown. The ROC curve plots the spam catch rate and the false positive rate obtained for each possible choice of decision threshold. The ROC curves were generated using a hand-labelled corpus of 176,000 messages of which roughly 120,000 were spam and roughly 56,000 messages were good mail.

The first algorithm shown in Figure 1 is Plagiarism Detection. This classifier compares incoming mail to a corpus of known spam and returns a score that indicates the degree of match between that message and the most similar spam email in the corpus. The second algorithm, SMTP Path Analysis, gathers routing information from a message's received lines and assigns a score based on the amount of spam that previously passed through the gateways used to send the message. The third classifier is a standard naïve-bayesian classifier.

Each of these classifiers has different strengths and demonstrates different degrees of effectiveness, as can be seen in the ROC curves.

Figure 1 also shows the results for the aggregate classifier created by combining these classifiers using equal weights. The graph shows that equal weights outperforms each of the individual classifiers over most of the graph, with equal weights performing only slightly worse than the naïve-bayesian classifier over a very small portion of its ROC curve.

WEIGHT OPTIMIZATION

How much better a performance can be achieved by non-equal weights? This question can be answered by formulating the selection of weights as an optimization problem in which a set of weights must be found that maximizes performance on a set of training data. One way to model this optimization problem is as a set of simultaneous equations that can be solved using linear regression. For each example in the training data, create an equation that represents the calculation of the weight for that example:

$$C(x) = w_0 + \sum_{i=1..N} w_i * S_i(x)$$

where $C(x)$ represents the score that the scoring function should produce for example x. The ideal weighting assigns a score of 1000 to all spam email and a score of 0 to all good email. This weighting categorizes all training data perfectly with full confidence. Therefore, a good choice for $C(x)$ is:

$$C(x) = \begin{cases} 1000 & \text{if x is spam} \\ 0 & \text{otherwise} \end{cases}$$

The above equations do not have an exact solution. Linear regression can find an approximate solution that minimizes the squared error between the ideal value of $C(x)$ and the value computed using the selected weights. Figure 1 also shows the performance of linear regression, which outperforms equal weights over most of the curve. The performance difference is negligible when the false positive rate is greater than 0.001. For false positive rates between 0.001 and 0.0002, linear regression offers as much as a 0.05 higher catch rate. While its performance does drop below equal weights just above the 0.0001 false positive rate, this is likely to be an artifact of the test corpus being too small to measure false positive rates accurately below 0.0005.

DYNAMIC AGGREGATION

The main limitation of linear regression is that it may assign a large weight to the best overall classifier, thus making the combined classifier sensitive to attacks against that classifier. This problem can be alleviated by updating the classifier weights dynamically in real time based on actual classifier performance. Then, if one classifier is attacked successfully, the weight for that classifier can be reduced to maintain overall performance. Dynamically updated weights provide even better resilience to attack than equal weights. Equal weight aggregators mainly help prevent individual attacks from succeeding. Continuously learned weights can, in addition, maintain high performance levels in the presence of one or more successful attacks by redistributing weights to the classifiers that remain effective.

Two small changes are recommended to manage the computational cost of linear regression in a dynamic

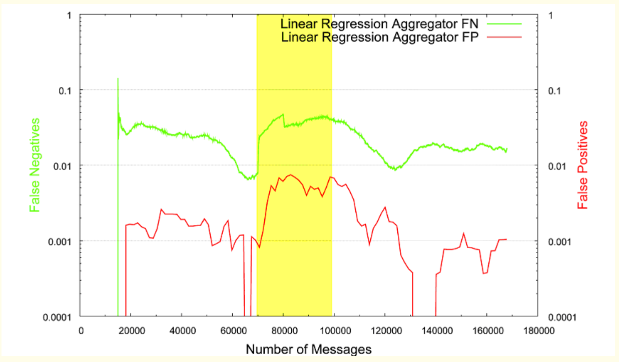


Figure 2. Performance of linear regression aggregator over time.

environment. First, rather than training on all available data, a sliding window can be used to limit the number of equations needed to a reasonable fixed size. Second, the frequency with which the weights are updated can be adjusted to balance the speed of adaptation with the computational cost of calculating the weights more frequently.

Figure 2 shows the performance over time of the linear regression aggregator. The aggregator was run with a sliding window of 5,000 examples and the weights were updated after every ten training examples. Each point in the graph was calculated using a moving average of the last 10,000 examples.

The graph shows that the false positive rate of this linear regression aggregator stays at or below 0.003 through the first 70,000 examples. The spam catch rate starts at around 0.90 and improves to around 0.98 at the 70,000 examples mark.

From 70,000 to 100,000 examples an attack was simulated against the naïve-bayesian classifier. The simulated attack modified half the scores returned by the naïve-bayesian classifier to one quarter of their original value. As a result, its catch rate drops below 0.50 during the attack. The linear regression aggregator notices this poor performance and lowers the weight of the naïve-bayesian classifier accordingly. This adjustment enables the overall system to maintain respectable performance during the attack. When the attack completes after 100,000 examples, the weight of the naïve-bayesian classifier is increased and the overall performance returns to its pre-attack levels.

CONCLUSION

The use of multiple classifiers can improve both the effectiveness and the accuracy of a spam filter while simultaneously increasing its resilience to attack.

One of the simplest and most effective ways to combine classifiers is a weighted average with equal weights assigned to each classifier. Equal weight aggregators can extract most of the potential from each classifier with little implementation or computational cost. Linear regression can be used to find a set of weights that outperforms equal weights, but the weights found may put too much onus on a single classifier and thus make the overall system susceptible to attack.

To get the best possible performance and resilience to attack, a dynamic aggregator can be used that adjusts its weights in real time based on actual classifier performance. The resulting spam filter achieves high performance levels and can redistribute its weights automatically as needed to adapt to attacks or long-term changes in the distribution of messages.

SUMMARY

ASRG SUMMARY: JANUARY 2005

Helen Martin

After a notably slow period towards the end of 2004, activity on the ASRG mailing list was revived with a veritable deluge of postings over the holiday season.

Gordon Peterson made it clear that he favours a fine-grained permissions-based approach to pursuing the spam problem, arguing that a fine-grained permissions list which, by default, does not allow HTML or attachments from untrusted senders, will virtually eliminate email as an effective vector for recruiting zombie spambot armies. Furthermore, he asserted that making the acceptance of HTML email contingent upon the sender having been whitelisted by the recipient would force spammers to abandon many of their content obfuscation tricks – thus making spam easier to identify and filter and resulting in reduced payback to spammers.

Jed Margolin suggested that email should be changed so that the content of an email resides on the sender's server until it is retrieved by the recipient – shifting the storage costs to the sender. Seth Breidbart pointed out that, while this would work for some situations, it would not work for legitimate senders without full-time servers. However, Gordon Peterson felt that the idea of shifting the costs and responsibility to the sender is not unreasonable (in principle). He thought that the problem of how a recipient would identify themselves to the sender's message storage server was an interesting one to consider.

Peter Kay posed the reflective question to the group 'Why are we still here?', but far from being a philosophical musing, his question was a serious one. He explained, 'all the anti-spam vendors ... claim high-9 catch rates and near-zero false positives. If that is the case, why are we still here trying to eliminate spam when one would think that, if the claims were true, the spam problem has been solved?'

der Mouse said that, while the claims made by some vendors may be fabricated, others are true – but only for a particular kind of user, particular kind of site, and so on. Phillip Hallam-Baker suggested that the simple answer is that filtering is 'only a tactical palliative and the criminal spammers have not been put out of business'. Danny Angus said that the goal must be to evolve a system which reduces the quantity of spam that is *sent*, rather than simply reducing the quantity that reaches its destination. Finally, James Lick answered simply '[We are still here] because there's still work to be done, and if we stand still spammers will figure out how to defeat current technology.'

[*A full archive of the ASRG mailing list can be found at http://www1.ietf.org/mail-archive/web/asrg/current/.*]