APRIL 2005

# virus BULLETIN

The International Publication on Computer Virus Prevention, Recognition and Removal

## CONTENTS



ISSN 0956-9979

## IN THIS ISSUE


### 1 NEW WORM RECEIVED

SymbOS/Commwarrior.A is the first worm to use MMS technology to spread on mobile phones. Will MMS become the replication method of choice among malware for mobile phones? Peter Ferrie and Frédéric Perriot fear that this might be the case.
**page 4**

### WHAT'S UP DOC?

Static analysis is a critical component of anti-virus strategies, but obfuscation techniques make it difficult to identify the calls made by malicious programs. Eric Uday Kumar, Aditya Kapoor and Arun Lakhotia present DOC, a tool for detecting obfuscated calls and returns in binaries.
**page 7**

### HATS OFF

As *Linux* makes gradual headway in the operating system battleground, *VB* continues to see a rise in the number of products submitted for *Linux* comparative reviews. This time there are 17.
**page 13**

### vbSpam supplement

This month: anti-spam news & events; Bayesian Noise Reduction; ASRG summary.

*'Ironically it is the simplest malware that takes 10 copies to trigger, while more complex malware ... is easier to detect.'*

**Alex Shipp, MessageLabs, UK**

## OUTBREAK DETECTION FROM THE TRENCHES

I was interested to read Oren Drori's article on outbreak detection in the March issue of *Virus Bulletin* (see *VB*, March 2005, p.9). Oren talks about the subject as if it were a theoretical concept, difficult to implement, but which might bear fruit if anyone got round to applying the concepts. As an AV researcher who has actively been researching and implementing outbreak detection since 1999, I thought readers of *VB* might be interested in the view from the trenches, and some firm facts and figures.

Oren mentions that, in order to achieve outbreak detection, the live email stream needs to consist of tens of millions of messages, over a wide geographical spread. However, my figures show this not to be the case. When I first started my research, I had access to traffic information on about 100,000 emails a day, all to destinations in the UK. Now, this has grown to 100,000,000 emails a day, all round the world. But comparing the data from then and now does not show any significant advantage in customer protection. When you think about it, this is obvious. Let us say a virus breaks out in Japan, but you do not have any Japanese customers. The virus does not appear on your 'outbreak radar', and so your customers are 'not protected'. But wait! The virus does not yet 'exist' for your customers because it is still only in Japan. Once it moves across the borders it appears on the radar; detection kicks in and customers are protected. So the size of the customer base, beyond some small critical mass, does not matter, because once the malware starts to impinge on those customers, customers become protected.

How many samples does it take to identify an outbreak? I would hazard a guess that any readers who are inexperienced in this field would estimate somewhere in the thousands or even tens of thousands. They may be surprised to learn that efficient outbreak detection kicks in at somewhere between two and ten copies.

How is this achieved? Well, of the 100,000,000 emails my engines are considering each day, only about 2000 to 4000 contain new objects which could potentially cause an outbreak. This is a very manageable number, and lends itself very well to extreme number crunching.

With over five years of historical data, it has been possible to do a lot of work on tuning and refining algorithms. Ironically, it is the simplest malware that takes 10 copies to trigger, while more complex malware which the authors have tried to hide by using polymorphic techniques is easier to detect because of the extreme unlikelihood of seeing two new objects in a time period where some characteristics are different, but others are the same.

These figures back up Oren's assertion that outbreak detection is a powerful complement to other types of protection. Gabor Szappanos presented a paper at VB2004 in which he asserted that if signature distribution could be cut down to around three hours, then mass-mailed malware would essentially be eliminated. Outbreak detection kicks in within minutes, and is well under this threshold.

Is catching the second copy the best we should be aiming for? After all, this means that only one customer is affected. By the definition of outbreak heuristics, catching the second copy is the best you can achieve using this type of protection. However, the infrastructure that needs to be put in place to perform traffic heuristics also lends itself to many other types of heuristics, and to performing the kind of analysis unimaginable on the desktop.

This is where the cutting edge of anti-virus research is going on, and it is a very exciting and fast-paced field. Perhaps this is a topic for another day, but I will leave you with this thought: first copy detection of most of the malware currently in the wild is comparatively easy with this kind of computing power available.

# NEWS

## US ANTI-SPYWARE BILL APPROVED

A revised anti-spyware bill was approved by a committee in the US House of Representatives last month.

The 'Securely Protect Yourself Against Cyber Trespass Act' (HR29) requires spyware programs to be both easy to identify and easy to remove, and restricts the collection of personal information to instances when express permission has been given by users. In addition, the penalties for those who violate the regulations have been stepped up, with the introduction of fines of up to $3 million per violation.

An amendment to the bill exempts cookies from the definition of spyware that is covered by the bill, as well as exempting embedded ads on web pages from the requirement that online ads include identifying information so users can find and remove the software causing them. Revised wording in the bill clarifies that companies will be allowed to monitor visitor activity on their own websites, and direct advertising of their own products (only) based on that monitoring.

The bill received unanimous approval from the Commerce Committee and is hoped to pass the full Congress this year.

## IT SECURITY 'MORE STRESSFUL THAN DIVORCE'

Keeping Europe's businesses free from viruses, Trojans, spam, phishing attacks and spyware is more stressful than going through a divorce, according to a survey of European IT security chiefs. A survey commissioned by *Websense* questioned technology managers at 500 European businesses about their experiences. 72 per cent of those questioned said that if they let their firm fall victim to malicious code their job would be on the line, and around 20 per cent said they felt that the responsibility of protecting their employer's business against hi-tech threats was more stressful than getting married, moving house or divorcing.

## ANTI-HYPE SITE GOING FOR A SONG

Industry hype-fighting website *VMyths* went up for auction last month on *eBay*. For ten years *VMyths* has prided itself on being the 'voice of reason' in the AV industry, debunking virus-related myths perpetuated by the media and slamming the hype generated by many anti-virus firms. *VMyths* owner Eric Robicheaud put the site up for sale along with the rights to an exclusive contract with editor Rob Rosenberger, whose forthright prose and acerbic wit have set the tone of the website, entertaining, informing and rattling cages in almost equal measures. The starting price for the site was set at $200,000, but at the time of writing no bids had been received.

### Prevalence Table – February 2005

| Virus | Type | Incidents | Reports |
|---|---|---|---|
| Win32/Netsky | File | 47,485 | 54.32% |
| Win32/Bagle | File | 19,451 | 22.25% |
| Win32/Sober | File | 13,765 | 15.75% |
| Win32/Mydoom | File | 1,397 | 1.60% |
| Win32/Bagz | File | 1,389 | 1.59% |
| Win32/Zafi | File | 782 | 0.89% |
| Win32/Dumaru | File | 570 | 0.65% |
| Win32/Klez | File | 362 | 0.41% |
| Win32/Mabutu | File | 327 | 0.37% |
| Win32/Lovgate | File | 217 | 0.25% |
| Win32/Funlove | File | 210 | 0.24% |
| Win32/Bugbear | File | 171 | 0.20% |
| Win32/MyWife | File | 147 | 0.17% |
| Win32/Valla | File | 140 | 0.16% |
| Win32/Mimail | File | 102 | 0.12% |
| Win32/Pate | File | 80 | 0.09% |
| Win32/Swen | File | 80 | 0.09% |
| Redlof | Script | 77 | 0.09% |
| Win32/Fizzer | File | 71 | 0.08% |
| Win32/Mota | File | 67 | 0.08% |
| Win95/Tenrobot | File | 62 | 0.07% |
| Win32/Yaha | File | 55 | 0.06% |
| Win95/Spaces | File | 40 | 0.05% |
| Win32/Sobig | File | 39 | 0.04% |
| Win32/Hybris | File | 37 | 0.04% |
| Win32/Magistr | File | 27 | 0.03% |
| Win32/BadTrans | File | 26 | 0.03% |
| Win32/Kriz | File | 20 | 0.02% |
| Win32/Plexus | File | 15 | 0.02% |
| Win32/Buchon | File | 14 | 0.02% |
| Win32/Nachi | File | 14 | 0.02% |
| Laroux | Macro | 13 | 0.01% |
| Others[1] | | 161 | 0.18% |
| Total | | 87,413 | 100% |

[1]The Prevalence Table includes a total of 161 reports across 56 further viruses. Readers are reminded that a complete listing is posted at http://www.virusbtn.com/Prevalence/.

# VIRUS ANALYSIS

## PARADISE LOST

*Peter Ferrie and Frédéric Perriot*
Symantec Security Response, USA

Eight months ago, Peter Ferrie and Péter Ször asked at the end of their article on SymbOS/Cabir: 'What will be next? A mass mailer using MMS?' (see *VB*, August 2004, p.4). The answer is yes, that is exactly what came next.

SymbOS/Commwarrior.A is the first worm to use MMS (Multimedia Messaging Service) technology to spread on cellular phones. Following in the footsteps of Cabir, it also replicates using Bluetooth, though with some improvements in its implementation. This double-pronged approach to replication makes Commwarrior a more likely candidate to be seen in the wild – although, at the time of writing, no such reports have been received.

As with Cabir, Commwarrior replicates only on *Nokia Series 60*-compatible devices.

### CONSULTING ON THE SUM OF THINGS

The worm begins by counting the number of copies of its process that are running. Generally, it will exit if another copy is running already (unless many copies start at the same time).

Next, the worm retrieves the machine identification number, and calculates an additive sum of the characters, to produce a unique value. This value might have been used by the worm's author during testing to avoid the infection of his own device, but now the result is simply discarded.

### TO NONE ACCOUNTABLE

The worm walks the list of running processes, and renames itself to the name of the first process in the list (which is usually 'EKern', the system kernel), followed by some random numbers.

In addition, the worm changes its owner and type to those of the first process's owner and type. Finally, the worm protects its process to prevent any other process from changing the priority of, or terminating, its process. Any attempt to terminate the process, by using a tool such as 'Switcher', is simply ignored. The screenshots in Figure 1 show the process list, with the legitimate EKern at the top of the list, and the worm process at the bottom of the list. Note the size difference.

If the worm has not been run from the 'c:\system\updates\commwarrior.exe' path, it creates the directories 'c:\system\updates' and 'c:\system\recogs', then

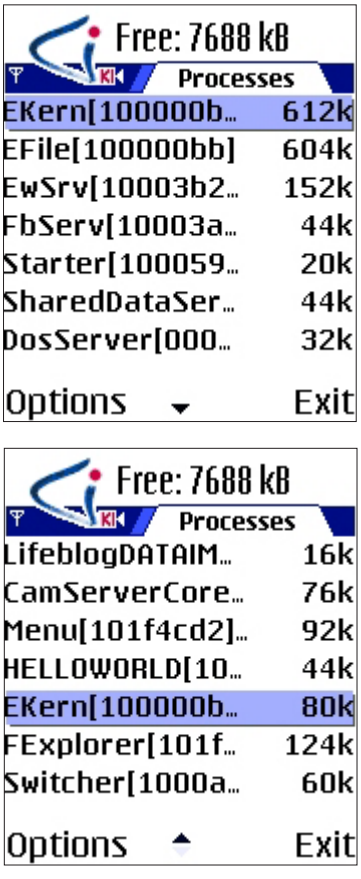



*Figure 1. The legitimate EKern process at the top, and the worm process at the bottom of the list.*

copies the 'commrec.mdl' file to the 'updates' and 'recogs' directories, and the 'commwarrior.exe' file to the 'updates' directory.

The 'commrec.mdl' file is a MIME recogniser file. It is intended to run the 'commwarrior.exe' file from the 'updates' directory whenever the phone starts, however on recent models of phones, such as the *Nokia 7610*, this does not work.

The worm creates a SIS file named 'c:\system\updates\commw.sis', by appending the 'commwarrior.exe' and 'commrec.mdl' files to the SIS header that is carried in its code. The SIS file uses the store method only – no compression is used – and the 'commwarrior.exe' file is marked to auto-execute on completion of the installation.

### AS SOFT AS NOW SEVERE, OUR TEMPER CHANGED

The worm contains various texts, but the most amusing is the one that reads 'OTMOPO3KAM HET' (which translates

roughly as 'No to softheads!'). According to our colleague Sergei Shevchenko, this is Russian slang, and the word for 'softheads' identifies someone whose brain has frozen so that the person has lost his ability to think and control himself.

## BOTH WHEN WE WAKE, AND WHEN WE SLEEP

The replication strategy of the worm is interesting because it adapts its infection vector according to the time of the day.

The worm uses Bluetooth during the normal waking hours of the phone's owner, when it is most likely to have other Bluetooth devices in range. It uses MMS during 'sleeping hours', and carefully cleans up the sent-message logs afterwards. In addition, the worm intentionally sets a lower priority to the replication threads, to make their activity less noticeable.

The overall scheduling of the worm's replication is accomplished by a single timer, which is set to trigger every ten seconds. Within the main timer callback, the worm checks for the payload condition, the time of day and the phone's Bluetooth state, in order to pick a replication method.

The worm favours finishing any on-going Bluetooth replication cycle over sending MMS messages. Its schedule looks like this:

- 08:00am – 11:59pm Bluetooth replication
- 12:00am – 06:59am MMS replication
- 07:00am – 07:59am MMS queue cleanup

## WHY HAS THOU ADDED THE SENSE OF ENDLESS WOES?

Commwarrior's payload triggers between midnight and 12:59am on the 14th day of any month. The worm's payload is to warm-boot the phone unconditionally.

Since the worm is part of the boot cycle, the phone could continue to reboot until the payload time ends.

## DIM ECLIPSE, DISASTROUS TWILIGHT

The Bluetooth replication code differs from the code seen in Cabir, in that it enumerates all the devices in range, whereas Cabir attempted to infect only the first device in range.

When the Bluetooth replication cycle starts, the worm enumerates all devices in range and builds a list of present devices. It queries each device for the availability of the 'Obex Push' service which is necessary to upload files. Devices meeting this condition are sent a copy of the SIS file of the worm, renamed to a random string which is eight characters in length, and consists of lower case letters and digits.

Once the worm has attempted replication to all devices that were found, it tears down all the connections and a new Bluetooth cycle can start.

The first Bluetooth cycle does not start until 50 seconds after the worm process starts, in order to let the phone boot completely. A new Bluetooth cycle can, in theory, be triggered every 50 seconds, but if there are many devices within range, it may be slower than that.

## RECEIVE THY NEW POSSESSOR

The worm's MMS functionality can be considered the equivalent of mass-mailing used by viruses on the *Windows* platform. This makes us very afraid that it will become the replication method of choice among any future self-replicating malware for cellular phones.

Commwarrior sends one MMS message at a time (i.e. one new MMS message at most every 10 seconds, since a message might take more than one cycle to complete). The recipients are picked randomly from the phone book.

On each cycle, one contact is picked at random, then Commwarrior enumerates the information fields of this contact, and selects from there the fields that correspond to mobile numbers only. This means that home numbers and work numbers (i.e. land line numbers) are ignored, in an attempt to maximise the chance of hitting other compatible cellular phones.

If a contact entry in the phone book contains several mobile numbers, then the MMS message is sent to all of those numbers.

From the debugging messages and code snippets in the Commwarrior code itself, it is possible to determine the origin of much of the MMS code used in the worm. Most of it was copied from a developer's page on a website, and altered slightly to add support for binary attachments (in fact, most of the rest of the code was copied too, from the *Symbian* SDK samples).

## IN THIS PERFIDIOUS FRAUD, CONTAGION SPREAD

For each of the MMS messages that Commwarrior sends, the subject and message body are chosen randomly from the following list:

Norton AntiVirus

Released now for mobile, install it!

Dr.Web

New Dr.Web antivirus for Symbian OS. Try it!

MatrixRemover

Matrix has you. Remove matrix!

3DGame

3DGame from me. It is FREE !

MS-DOS

MS-DOS emulator for SymbvianOS. Nokia series 60 only. Try it!

PocketPCemu

PocketPC *REAL* emulator for Symbvian OS! Nokia only.

Nokia ringtoner

Nokia RingtoneManager for all models.

Security update #12

Significant security update. See www.symbian.com

Display driver

Real True Color mobile display driver!

Audio driver

Live3D driver with polyphonic virtual speakers!

Symbian security update

See security news at www.symbian.com

SymbianOS update

OS service pack #1 from Symbian inc.

Happy Birthday!

Happy Birthday! It is present for you!

Free SEX!

Free *SEX* software for you!

Virtual SEX

Virtual SEX mobile engine from Russian hackers!

Porno images

Porno images collection with nice viewer!

Internet Accelerator

Internet accelerator, SSL security update #7.

WWW Cracker

Helps to *CRACK* WWW sites like hotmail.com

Internet Cracker

It is *EASY* to *CRACK* provider accounts!

PowerSave Inspector

Save you battery and *MONEY*!

3DNow!

3DNow!(tm) mobile emulator for *GAMES*.

Desktop manager

Official Symbian desctop manager.

CheckDisk

*FREE* CheckDisk for SymbianOS released!MobiComm

(Due to what appears to be a missing terminating character, the last message body appears to contain the subject ['MobiComm'] for the next message body ['MobiComm, Mobile communications inspector. Try it!'] which is never referenced.)

This worm's use of social engineering is very similar to that seen in many email worms, and has proven very successful in the past. The MMS messages contain an attachment whose name is always 'commw.sis'. The attachment is the worm installer, and its MIME type is set explicitly to 'application/vnd.symbian.install'.

The worm maintains a list in memory of all of the recipients of its MMS messages, and uses the list to avoid sending multiple messages to any recipient. In the event that the phone is switched off (or the payload executes), the list will be lost, and recipients will receive additional messages if the worm process is executed again.

In the early hours of the morning, the worm cleans up the MMS queue. This means that the user will not be alarmed by any worm messages in the 'Sent' box.

## JOURNEYED ON, PENSIVE AND SLOW

One mitigating factor to the success of the MMS replication method is that the phone operator interoperability seems to be very limited. Indeed, during our attempts to send our own test messages, we experienced many failures to send messages at all between different providers, and long delivery times.

It should be noted that, upon receipt of the SIS file, whether by Bluetooth or MMS, the user must agree explicitly to its installation via several dialog boxes. If, at any point, the user cancels the installation, the worm does not execute.

## CONCLUSION

Due to its openness and the ready availability of development tools, the *Symbian* platform appears to be a fertile ground for new malware, and will become a required area of expertise for current and future anti-virus researchers. The fact that the *Symbian* OS is designed to run on embedded platforms, whose resources are limited, and that its core APIs are based on C++, can throw off reverse engineers who are used to the PC platform.

# TECHNICAL FEATURE

## DOC – ANSWERING THE HIDDEN 'CALL' OF A VIRUS

*Eric Uday Kumar, Aditya Kapoor, Arun Lakhotia*
University of Louisiana at Lafayette, USA

Malicious programs use obfuscations to hide information about the system calls they make. Detector of Obfuscated Calls, or DOC, is a prototype tool which demonstrates a technique for detecting obfuscated calls and returns in binaries. DOC identifies several types of obfuscations statically, promising to speed up the process of determining whether or not a program is malicious.

### INTRODUCTION

One of the first steps in determining whether a program is malicious is to identify the system calls it makes. If the program performs certain collections of file operations, registry operations, or network operations, there may be good reason to consider it likely to be malicious.

The set (or sequence) of system calls a program makes is referred to as its behaviour. The behaviour of a program may be determined either by static analysis or by dynamic analysis.

In static analysis, a program is analysed (by humans and/or tools) without running or simulating it. In dynamic analysis, a program's behaviour is observed, often by trapping the calls or sniffing network activity.

Malware writers have developed obfuscation techniques that make it difficult, using static analysis techniques, to identify the calls made by their program. Effectively, these programs make a call without actually using the call instruction (see Peter Ször and Peter Ferrie, 'Hunting for Metamorphics', *Virus Bulletin Conference 2001*). Doing this increases the difficulty of analysing a program not least because it defeats the methods that typical disassemblers use to identify procedure entry and exit points.

Therefore, anti-virus companies tend to rely on dynamic methods for determining a program's behaviour. For instance, *Symantec*'s Bloodhound technology executes a program in a sandbox (or an emulator), traps the calls made by the program, and then determines whether or not it is malicious.

However, while dynamic analyses are helpful and often necessary, they have a tendency to be cumbersome, time-consuming and fallible.

Malware authors already have many methods for defeating detection through dynamic analysis, including detecting the dynamic analysis method, introducing delay loops to foil stopping heuristics, and executing their malicious behaviour only in particular circumstances.

For these reasons alone static analysis is still a critical component of anti-virus strategies, but methods for overcoming obfuscation obstacles are extremely desirable.

In this article we present the results obtained by using a new tool called DOC (Detector of Obfuscated Calls) to analyse the virus W32/Evol. DOC identifies statically several types of obfuscations related to the call and return instructions.

Technical details of the method used by DOC have been described elsewhere (Lakhotia and Kumar 2004, *Fourth IEEE International Workshop on Source Code Analysis and Manipulation*). In this article we will review call/return obfuscations, describe DOC and how it was applied to W32/Evol, and summarise some of the successes and limitations of the approach.

### CALL/RETN OBFUSCATIONS

Figure 1 shows a classic example of call obfuscation used by viruses, most notably W32/Evol and Netsky.Z.

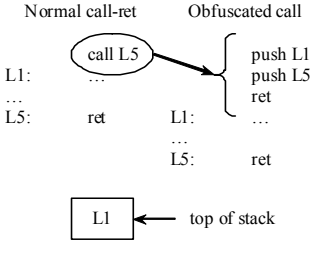



*Figure 1. Call obfuscation.*

In the left-hand column is a normal call instruction. In the right-hand column is code containing a sequence of two push instructions and a retn instruction. These three instructions do exactly the same work as the call instruction. They are semantically equivalent.

Other related obfuscations include the substitution of retn instructions and the use of non-contiguous function bodies. For instance, a retn instruction may be replaced by the instruction sequence: pop <reg> followed by jmp <reg>, where <reg> is any general purpose register. Non-contiguous procedure bodies can be created by intertwining a procedure's code with the code of other procedures, thus making it difficult to match a call instruction to its corresponding retn instructions.

Such obfuscations take away important cues that are used during both automated and manual analysis. While a determined, experienced programmer would be able to discover the obfuscations, the time that it takes to make the discovery can be very precious when the malware is spreading actively.

Substituting call instructions, in particular, breaks most automated methods for detecting a virus since these

methods depend on recognizing call instructions both to identify the kernel functions used by the program and to identify procedures in the code. As is shown later, *IDA Pro*, a disassembler used very widely in the anti-virus industry, gives incorrect and misleading results in the presence of call/return obfuscations.

## ABOUT DOC

DOC is implemented in Java as a plug-in to the Eclipse Platform (see http://www.eclipse.org/). Figure 2 shows a screenshot of DOC when opening an assembly file (.asm extension).

DOC allows any number of projects to be opened at the same time. The navigator view (on the left-hand side) is used to browse and open files in a project. The files are displayed in the file view (shown on the right).

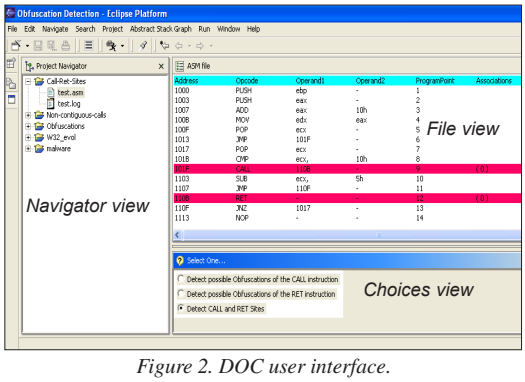



*Figure 2. DOC user interface.*

DOC takes as its input an assembly file or a disassembled binary obtained from a disassembler such as *IDA Pro*. The user may select any of the following three analyses:

- Match call-ret instructions
- Detect obfuscated calls
- Detect obfuscated returns

DOC returns its results by highlighting and annotating the assembly. The annotations contain links to related code when there are multiple occurrences of the same type of obfuscation.

## INSIDE DOC

DOC uses abstract interpretation, a technique commonly used in static analysis. In this technique a program is interpreted using abstract values, instead of real values. The key challenge in using abstract interpretation is in choosing the right abstraction. DOC creates an abstraction of the stack and its contents.A specific instance of a real stack is represented as an abstract stack.

Further, the set of all possible abstract stacks for all possible executions of a program is represented as an abstract stack graph. Although the set of all abstract stacks (or real stacks) for all possible executions of a program may be infinite, the abstract stack graph is finite.

The abstract stack graph for a given assembly program is constructed by interpreting each instruction of the program. The operations performed by the instruction on a real stack are performed instead on an abstract stack graph. Each instruction is interpreted at most once.

Once the abstract interpretation terminates, the abstract stack graph contains an abstraction of all possible stacks at each statement. DOC analyses the abstract stack to match call-ret instructions, detect obfuscated calls, and detect obfuscated returns.

## W32/EVOL – REVEALING THE HIDDEN

It was our efforts at analysing W32/Evol statically that led us to develop DOC. It all started a few years ago as a result of our first attempt at developing an anti-virus scanner based on formal, static analysis. We had implemented a behaviour-based analyser using model checking – however, our analyser failed miserably when we exposed it to W32/Evol.

A closer analysis revealed that the virus was obfuscating all system calls, and our analyser made the assumption that *IDAPro* would detect system calls correctly in disassembled code. It failed and, as is so common in developing new technologies, its failure provided the impetus to explore new methods.

Here we describe some of the causes for disassembly failure and show how DOC can detect these.

### Call/ret obfuscation in W32/Evol

The common sequence of instructions to make a system call (for example GetTickcount) in a *Windows* environment is as follows:

```
push   add1   ; "kernel32.dll"
push   add2   ; "GetTickCount"
call   GetProcAddress
call   [eax]  ; "call GetTickCount"
```

Here, 'add1' and 'add2' are pointers to the strings 'kernel32.dll' and 'GetTickCount' located in the data segment. The addresses of these strings are pushed on the stack.

The kernel32.dll function GetProcAddress is called, which returns the address of the function 'GetTickCount' in the

```
:0040153F                mov     dword ptr [eax], 'TteG'
:00401545                mov     dword ptr [eax+4], 'Ckci'
:0040154C                mov     dword ptr [eax+8], 'tnuo'
:00401553                mov     byte ptr [eax+0Ch], 0
:00401557                push    eax
:00401558                call    sub_401280
:0040155D                push    eax
:0040155E                call    sub_4012A7
:00401563                mov     [ebp+0], eax
:00401566                add     esp, 10h
:00401569                pop     ebp
:0040156A                retn
:0040156A sub_401530     endp ; sp = -0Ch
```

*Figure 3. W32/Evol code with multiple obfuscations.*

eax register. The program then does an indirect call to the address in eax, effectively making a call to GetTickCount.

Disassemblers such as *IDA Pro* can detect such call patterns and aid in detecting system calls. Figure 3 shows a code fragment from W32/Evol for calling the function GetTickCount. This code has multiple obfuscations, none of which are detected by *IDA Pro*. The reasons for this are instructive.

*IDA Pro* assumes that the retn instruction at address 0040156A actually returns from the procedure. Thus, it deems this statement as ending the procedure that has an entry at address 00401530.

*IDA Pro* indicates the end of a procedure by introducing the dummy directive endp. Thus it deduces that the retn statement matches 'call 00401530' instructions.

The retn instruction, it turns out, is performing a call. The value returned from GetProcAddress is moved to the stack, and the stack pointer is modified such that when the retn instruction is executed, it transfers control to GetTickCount.

```
0040153F mov dword ptr ds:[eax], 54746547 ;'TteG'
00401545 mov dword ptr ds:[eax+4], 436B6369 ;'Ckci'
0040154C mov dword ptr ds:[eax+8], 746E756F ;'tnuo'
00401553 mov byte ptr ds:[eax+c], 0;  '\0'
00401557 push eax; ptr to "GetTickCount".
00401558 call 00401280; gets base address of
                        kernel32.dll base.
0040155D push eax
0040155E call 004012A7; obfuscated call to
                        GetProcAddress()
00401563 mov dword ptr ss:[ebp], eax; addr of
                                      GetTickCount().
00401566 add esp, 10
00401569 pop ebp
0040156A retn; transfer control to GetTickCount().
```

*Figure 4. Annotated code of Figure 3.*

This can be verified by analysing the virus manually in a debugger such as *OllyDbg*.

Figure 4 presents the code of Figure 3 with annotations created by such a manual analysis.

## Detecting call obfuscations

Figure 5 shows a portion of the code where DOC detects the obfuscated call to the kernel function GetTickCount().

| | | | | | |
|---|---|---|---|---|---|
| 0040153F | MOV | DS:[EAX], | 54746547 | 443 | |
| 00401545 | MOV | DS:[EAX+4], | 436B6369 | 444 | |
| 0040154C | MOV | DS:[EAX+8], | 746E756F | 445 | |
| 00401553 | MOV | DS:[EAX+C], | 0 | 446 | |
| 00401557 | PUSH | EAX | - | 447 | (0) |
| 00401558 | CALL | 00401280 | - | 448 | |
| 0040155D | PUSH | EAX | - | 449 | |
| 0040155E | CALL | 004012A7 | - | 450 | |
| 00401563 | MOV | SS:[EBP], | EAX | 451 | |
| 00401566 | ADD | ESP, | 10 | 452 | |
| 00401569 | POP | EBP | - | 453 | |
| 0040156A | RETN | - | - | 454 | (0) |

*Figure 5. Using DOC to detect obfuscated call.*

The push instruction at address 00401557 and the retn instruction at address 0040156A are instrumental in obfuscating the call to GetTickCount(). This is indicated by highlighting these instructions. The annotation '(0)' at the end of these instructions indicates that the two belong to the same call obfuscation.

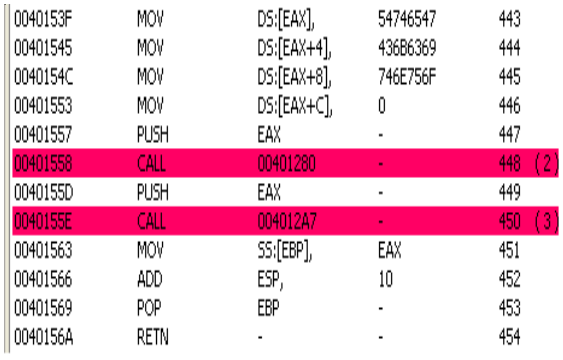
W32/Evol uses similar code to make system calls in 25 locations. *IDA Pro* misses all of these calls, whereas DOC highlights every such retn instruction as making a call.

## Matching call-retn instructions

Figure 6 shows the same code as that shown in Figure 3, but it also shows the results of running DOC's analysis for matching call-retn instructions.

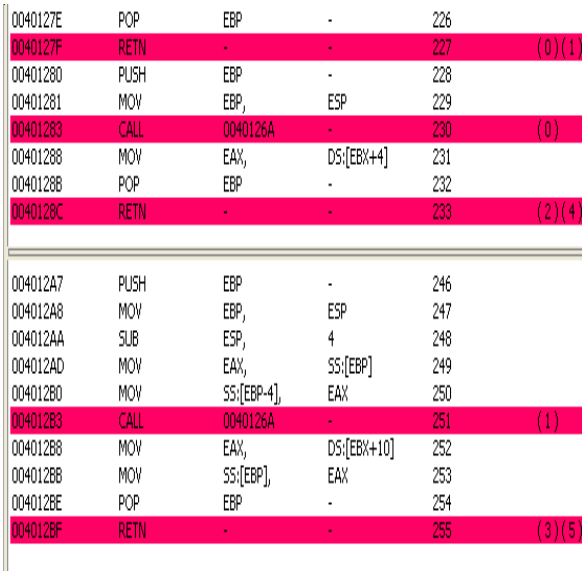
The two call instructions at addresses 00401558 and 0040155E are highlighted and are annotated '(2)' and '(3)', respectively. These numbers are arc labels in the effective call graph.

Figure 7 shows return sites corresponding to these statements. These statements are annotated with the numbers '(2)' and '(3)', which are matched to the call sites so labelled. This figure also shows retn statements matching call sites annotated as '(0)' and '(1)'. As is expected, one retn statement may match multiple call sites. DOC correctly found matching retn statements for all 33 call statements of



| | | | | | |
|---|---|---|---|---|---|
| 0040153F | MOV | DS:[EAX], | 54746547 | 443 | |
| 00401545 | MOV | DS:[EAX+4], | 436B6369 | 444 | |
| 0040154C | MOV | DS:[EAX+8], | 746E756F | 445 | |
| 00401553 | MOV | DS:[EAX+C], | 0 | 446 | |
| 00401557 | PUSH | EAX | - | 447 | |
| 00401558 | CALL | 00401280 | - | 448 | (2) |
| 0040155D | PUSH | EAX | - | 449 | |
| 0040155E | CALL | 004012A7 | - | 450 | (3) |
| 00401563 | MOV | SS:[EBP], | EAX | 451 | |
| 00401566 | ADD | ESP, | 10 | 452 | |
| 00401569 | POP | EBP | - | 453 | |
| 0040156A | RETN | - | - | 454 | |

*Figure 6. Using DOC to detect valid calls.*



| | | | | | |
|---|---|---|---|---|---|
| 0040127E | POP | EBP | - | 226 | |
| 0040127F | RETN | - | - | 227 | (0)(1) |
| 00401280 | PUSH | EBP | - | 228 | |
| 00401281 | MOV | EBP, | ESP | 229 | |
| 00401283 | CALL | 0040126A | - | 230 | (0) |
| 00401288 | MOV | EAX, | DS:[EBX+4] | 231 | |
| 0040128B | POP | EBP | - | 232 | |
| 0040128C | RETN | - | - | 233 | (2)(4) |
| 004012A7 | PUSH | EBP | - | 246 | |
| 004012A8 | MOV | EBP, | ESP | 247 | |
| 004012AA | SUB | ESP, | 4 | 248 | |
| 004012AD | MOV | EAX, | SS:[EBP] | 249 | |
| 004012B0 | MOV | SS:[EBP-4], | EAX | 250 | |
| 004012B3 | CALL | 0040126A | - | 251 | (1) |
| 004012B8 | MOV | EAX, | DS:[EBX+10] | 252 | |
| 004012BB | MOV | SS:[EBP], | EAX | 253 | |
| 004012BE | POP | EBP | - | 254 | |
| 004012BF | RETN | - | - | 255 | (3)(5) |

*Figure 7. Using DOC to detect valid call-ret sites.*

W32/Evol. In several instances the procedure code was not contiguous.

## CONCLUSIONS

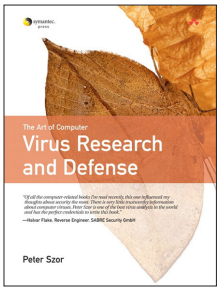
DOC is efficient, being linear in both space and time. And it is demonstrably effective in finding the sort of call/retn obfuscations found inW32/Evol. We believe its techniques could become an important part of an anti-virus researcher's toolkit, and that they can significantly speed up analysis of obfuscated binaries. DOC does have a number of limitations. It is restricted solely to detecting call obfuscations, and cannot handle some of these, including manual stack manipulation. Efforts to overcome some of these limitations are currently in progress in our laboratory. [*For more details of the authors' research see http://www.cacs.louisiana.edu/~arun/home/index.html.*]

# BOOK REVIEW 1

## THE ART OF DEFENCE

*Vanja Svajcer*
SophosLabs, UK

**Title:** The Art of Computer Virus Research and Defense
**Author:** Péter Ször
**Publisher:** Addison Wesley for Symantec Press
**ISBN:** 0-321-30454-3




It has been more than six years since I started working as a virus researcher, but I remember the first few months vividly. The beginning of any job is difficult, but even more so if you have to acquire your skills using a number of highly scattered, incomplete and sometimes suspicious resources.

As a beginner, I was surprised and disappointed to find out that there were very few books on the subject of computer viruses. Furthermore, none of the books were dedicated to people who, like me, were eager to dig into the low-level technical issues of viruses and the technology required to tackle them. Some books, like Fred Cohen's *A Short Course on Computer Viruses*, were intriguing but I felt I needed something more practical – a proper handbook to point me in the right direction. Unfortunately, for me, there was no choice but to learn the hard way.

It was with great excitement, therefore, that I learned recently that Péter Ször's book *The Art of Computer Virus Research and Defense* was to be published. I pre-ordered a copy straight away and waited for what seemed like forever to receive it.

Although I had not known what to expect, my first encounter with the book reassured me that I would not be disappointed. The book weighs in at 675 pages and is the result of one year's work – which is even more impressive considering the fact that it was written mainly during weekends. Behind the book's impressive content is not just Péter's 15-plus years of expertise, but also a breadth of knowledge gathered from many of the best-known members of the anti-virus research community.

The book is divided into two parts and 16 chapters. The two parts, 'Strategies of Attacker' and 'Strategies of Defender', are dedicated to a specific set of problems but there are many occasions where this division blurs. The title 'Strategies of Attacker' may sound a little confusing, but the content is always written from the point of view of a defender and very few ideas are exposed that could be used by a malicious reader. At the end of every chapter is a very useful list of references for those who need to know more.

The introductory chapters are clear and well structured, but I felt that they could have been a little gentler for beginners. Although the book states that the reader is expected to have a programming background, a chapter containing an introduction to the CPU architecture, assembly language and operating system would have been a beneficial addition for the less experienced reader – without this, some of the early assembler examples could prove discouraging.

The first part of the book shines as this is where we find Péter's best known work – the technical details on Win32 threats, vulnerabilities and exploits, worm analysis and particularly interesting coverage of polymorphism and metamorphism. This part of the book is the most valuable to any reader who is keen to learn as much as possible about current viral threats and the technology used by the virus-writing community. Here the majority of content is derived from articles published in *Virus Bulletin* or papers presented at various conferences. It is certainly a good thing to find all that work in one place but I felt that in some cases the otherwise natural flow of the book was interrupted.

The second part of the book is a pleasant surprise and demonstrates Péter's intimate knowledge of the internals of many anti-virus and security products and technologies. This part of the book is an excellent source of information when one needs to explain the fact that modern anti-virus software uses a set of increasingly sophisticated methods for virus detection. The chapter containing an introduction to anti-virus technology is followed by detailed explanations of problems and solutions for handling *Windows* memory scanning and disinfection, as well as deep insights into generic blocking techniques and network level defence.

The 'Strategies of Defender' section contains a very useful chapter on analysis techniques. The chapter also gives an overview on how to set up a virus analysis laboratory. Although the book provides a good level of detail I felt that this subject should span more than one chapter. One would guess that time constraints prevented further coverage of this subject, and hope that the content will be expanded in the next edition(s). Another useful addition to the book would be a CD-ROM containing the tools used to analyse viruses and perhaps some demonstration programs that the reader could use to practise the analysis.

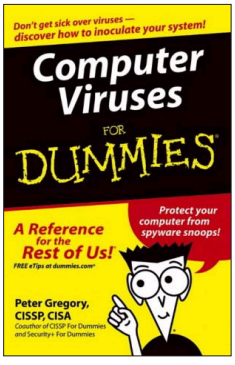
My only objection concerns the title of the book, which suggest that its scope is as majestic as Knuth's *The Art of Computer Programming*. Even with twice as many pages and double the content the book would not be able to reach the depth required to describe the subject fully. For me, a better title would have been 'An Overview of the Art of Computer Virus Research and Defense'. However, this does not prevent Péter's book from being, in my opinion, the best book on the subject published to date.

# BOOK REVIEW 2

## DUMMIES' GUIDE TO VIRUSES

*Paul Baccas*
SophosLabs, UK

**Title:** Computer Viruses for Dummies
**Author:** Peter Gregory, CISSP, CISA
**Publisher:** Wiley
**ISBN:** 0-7645-7418-3




As a publishing phenomenon the 'for Dummies' series has run the gamut from A to Z over the academic and not so academic disciplines. Unfortunately, in running such a gamut you will perforce travel both through 'nadir' and 'zenith'. This tome leans heavily towards the former, thanks to a number of glaring errors.

My first complaint about this book is that the title is a misnomer. This is not a book about computer viruses *per se*, but rather a book about personal computer security for the home user. While, naturally, a great deal of the topic concerns computer viruses, the book does not inform the reader extensively about them.

Another serious error was made with the timing of the publication of this book. Whether by ignorance or design the publication date of August 2004 was unfortunate, since the book was able to make no mention of *Windows XP Service Pack 2* (which was also released in August 2004). While not a panacea, *SP2* has by its very nature changed the home computing market with its specific focus towards security.

Some parts of the book contain fabrications worthy of the most sordid tabloid journalist. In fact, the motto 'never let the facts get in the way of a good story' would be apt in many cases. A selection of howlers:

- 'Brain, the first virus'
- 'Concept virus was the first encrypted virus'
- 'Norton VirusScan was the first anti-virus program'

Part I of the book deals with assessing the risks that arise when a computer is connected to the Internet and describes how to combat them. The section begins with an explanation of viruses and other malware the computer user may encounter. Next, a general chapter describes what symptoms and changes a computer may exhibit if malicious code is running on it. These are followed by an introduction to finding, running and updating anti-virus solutions.

Part II is wholly concerned with anti-virus software. It begins by looking at how to evaluate, acquire and install anti-virus solutions. As part of the section on evaluating anti-virus products the book focuses on many functions of the anti-virus software – with the exception of virus detection. No mention is made of independent anti-virus testing, or even magazine reviews.

The next chapter looks at and explains some of the jargon involved in configuring anti-virus software. This is followed by a section that is best described as 'what to scan and when to scan it'. Finally, a chapter describes what to do if the software detects a virus. Importantly, this section tells the reader to find out what the virus has done before removing it.

Part III deals with the aspects of security software that are often neglected. Computer security is not like forestry where you plant a sapling and leave it – it is more like bonsai, where constant nurturing is required. This includes updating anti-virus data and programs, patching the operating system and applications, and running anti-spyware and firewall programs.

A chapter is devoted to PDAs and describes how they should be protected. Part III finishes with an overarching chapter on how to practise 'safe hex' – from using legitimate software to avoiding spam.

In my opinion Part IV lets the book down. The chapter on the history of viruses contains many errors (some of which were mentioned earlier). This is followed by a chapter on Trojans, worms, hoaxes, and spam – where there are more statements with which experts will disagree. Finally, there is an explanation of how viruses infect and virus taxonomy, which includes further interesting assertions such as: 'The other name for a macro virus is Trojan horse [*sic*].'

Part V, the ubiquitous 'part of tens', ends the book. There are two chapters here; the first concerns virus myths and the second concerns anti-virus programs. The virus myths range from 'anti-virus companies write viruses' to 'viruses broke my computer'. The last chapter lists ten anti-virus programs with a two-third page summary which lists manufacturer, website etc., along with a 'yes/no' list of features.

One would have to assume that the intended audience for this book is the 'average joe' computer user, but I cannot see why it was written – the book contains no new information and no new insights. What's more, the information the book provides can be gleaned from various other sources and publications, most of which are available free of charge. The author's website does not elaborate on the subject either – although it does provide the opportunity to purchase most of the software programs that are mentioned in the book.

# COMPARATIVE REVIEW

## RED HAT LINUX 9

*Matt Ham*

With *Linux* still making gradual headway in the operating system battleground it comes as little surprise that there are more products in this year's *Linux* comparative than the last, or that the products submitted this year are more feature-packed. On the last occasion 14 products were submitted; this time there are 17.

The additions to the line up for this test are: *Avira*, *MicroWorld*'s *eScan* and *Norman Virus Control*. All of these are from companies that are familiar with *VB*'s testing regime – indeed, *Avira* is developed by the same team that produces *H+BEDV*'s *Antivir*, so they have first-hand experience of testing on the *Linux* platform too.

In the last *Linux* test there were problems of a technical nature and problems that were more informational in nature. Technically, the on-access scanners were a very mixed bag, ranging from stable to likely to fall over at the drop of a pin. In last year's test neither the *Sophos* product nor the *McAfee* product had an on-access scanner. *McAfee* has since added this functionality, leaving *Sophos* as the odd man out in this year's review.

The majority of *Linux* products use *Dazuko* as their on-access scanning solution, which proved to be reliable last year. Twelve months on, even greater stability should be expected all round.

The second problem encountered in last year's *Linux* review related to updating the products, it not always being apparent how updates should be applied without direct access to the Internet. This is an increasing problem on all platforms since Internet access is considered to be a standard feature these days. Such reasoning can render it very difficult to update an isolated machine before connecting it to the net – clearly protection is required *before* connecting a machine to a resource that is plagued with a multitude of threats ready to attack a vulnerable machine. It seems quite common for developers to ignore this issue, however, so I was prepared for updating to be a major problem.

The other complaint arising from last year's *Linux* comparative concerned product documentation. Although still not ideal, the documentation seemed less problematic this time.

### TEST SETS

The test sets were updated to the latest WildList data available on 24 February 2005. In fact, this was the December 2004 data. The deadline for product submissions was 28 February 2005, meaning that the task ahead of the products was somewhat less than taxing, since their developers had each had a full two months to react to files submitted by customers or obtained from other developers. Additions to the WildList this time consisted of a further bunch of tedious worms and, again, these were not expected to present any significant challenge for the products.

### Alwil Avast 1.0.8.2

| | | | |
|---|---|---|---|
| **ItW File** | 100.00% | **Macro** | 99.56% |
| **ItW File (o/a)** | 100.00% | **Standard** | 99.36% |
| **Linux** | 50.00% | **Polymorphic** | 93.57% |


*Avast* is a *Dazuko*-based scanner as far as on-access functionality goes, and has moved from late beta to a fully released product over the last year. On this occasion the only major sticking point was the application of a licence file to the product which was not named as the product expected (due to *Linux* case-sensitivity), causing the *Avast* daemon to refuse to operate. Once this problem had been overcome, however, all was plain sailing. As far as installation was concerned *Avast* differed slightly from the majority of other products, in that it spread its files far and wide. Most installations in this test were located in the opt directory, which seems to be a *de facto* standard.

Problems encountered with *Avast*'s on-access scanner in previous reviews had vanished and detection rates were very similar to those obtained last year, so it is no surprise that *Alwil* receives another VB 100% award in this test.

### Avira Avira 1.1.3-17

| | | | |
|---|---|---|---|
| **ItW File** | 100.00% | **Macro** | 100.00% |
| **ItW File (o/a)** | 100.00% | **Standard** | 100.00% |
| **Linux** | 91.67% | **Polymorphic** | 100.00% |


As mentioned earlier, *Avira* is a 'new' product that does not really count as such, since its developer, *H+BEDV*, is an old hand at *VB* comparative testing and also very much connected with the *Dazuko* project. It comes as little surprise, therefore, that the on-access scanner is powered by this module. Installation of the product was easy and its detection was excellent – better even than the *Windows* product reviewed earlier this year (see *VB*, February 2005, p.12). The detection rate does come at a price though: this is one of the noticeably slower scanners in the line-up. There were no false positives to mar the performance and thus *Avira* receives a VB 100% award.

| On-access tests | ItW file | | Macro | | Polymorphic | | Standard | | Linux | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Number missed | % | Number missed | % | Number missed | % | Number missed | % | Number missed | % |
| Alwil Avast | 0 | 100.00% | 18 | 99.56% | 114 | 93.44% | 14 | 99.54% | 9 | 80.00% |
| Avira Avira | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 3 | 86.67% |
| CAT Quick Heal | 0 | 100.00% | 74 | 98.20% | 314 | 96.25% | 103 | 96.35% | 7 | 60.00% |
| Doctor Web Dr.Web | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 2 | 99.82% | 0 | 100.00% |
| Eset NOD32 | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% |
| F-Secure Anti-Virus | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 1 | 93.33% |
| FRISK F-Prot Antivirus | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 2 | 99.72% | 0 | 100.00% |
| Grisoft AVG | 0 | 100.00% | 0 | 100.00% | 425 | 83.72% | 42 | 97.33% | 16 | 48.33% |
| H+BEDV Antivir | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 3 | 86.67% |
| Kaspersky Anti-Virus | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% |
| MicroWorld eScan | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 1 | 93.33% |
| McAfee LinuxShield | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 1 | 99.91% | 0 | 100.00% |
| Norman Virus Control | 0 | 100.00% | 10 | 99.75% | 147 | 92.09% | 10 | 99.57% | 6 | 66.67% |
| SoftWIN BitDefender | 0 | 100.00% | 26 | 99.31% | 6 | 99.73% | 21 | 99.42% | 8 | 73.33% |
| Sophos SWEEP | - | - | - | - | - | - | - | - | - | - |
| Trend Micro ServerProtect | 0 | 100.00% | 0 | 100.00% | 215 | 95.77% | 10 | 99.53% | 7 | 65.33% |
| VirusBuster VirusBuster | 3 | 99.76% | 3 | 99.93% | 3048 | 77.13% | 68 | 97.12% | 37 | 26.67% |

## CAT Quick Heal X Gen 7.03

| | | | |
|---|---|---|---|
| **ItW File** | 100.00% | **Macro** | 98.20% |
| **ItW File (o/a)** | 100.00% | **Standard** | 96.33% |
| **Linux** | 60.00% | **Polymorphic** | 96.25% |

In last year's review only one product offered a GUI, so it came as something of a surprise to note that an increasing number of products in this test had GUI functionality. *CAT*'s product was the first of these. *CAT*'s GUI is QT-based and totally optional, and was not, therefore, used for scanning purposes. This was the default decision taken wherever a GUI could easily be avoided, since the majority of products are significantly easier to test from the command line.


That said, the results for ItW scanning were perfect, with just a few misses in the other test sets. The speed of scanning was also towards the faster end of the spectrum. *Quick Heal* gained a VB 100% on its first test on *Linux* last year, and easily obtains another on this outing.

## Doctor Web Dr.Web for Linux 4.32.2

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

*Dr.Web* is the first product in this review which does not use *Dazuko* for on-access scanning. Instead, it uses a vfs object called by the Samba daemon. Historically these solutions have been slightly prone to hiccups, although *Dr.Web* seems to have avoided these consistently. No problems of any type were noted during installation or operation, and all but two files on access were detected in the whole of the test sets. *Dr.Web* thus continues its good work and gains another VB 100% as a result.


## Eset NOD32 2.03

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

Having dabbled with kernel objects in the past, the *Eset* developers have now opted for the simpler life and use *Dazuko* for on-access scanning. The last test of this product on *Linux* demonstrated no technical problems but a whole host of different operations were required to install and configure the product. This has been simplified significantly, with one RPM file replacing the trickery that was required previously. The results of scanning were eminently predictable: all files were detected, with a VB 100% award the equally predictable result.


## F-Secure Anti-Virus 4.62

| | | | |
|---|---|---|---|
| **ItW File** | 100.00% | **Macro** | 100.00% |
| **ItW File (o/a)** | 100.00% | **Standard** | 100.00% |
| **Linux** | 93.33% | **Polymorphic** | 100.00% |

*F-Secure Anti-Virus* (*FSAV*) proved to be a very frustrating beast initially, with all attempts to tame it failing dismally. However, this changed instantly when it became apparent that there are two copies of the configuration file for the product. Altering one set seems to have no effect whatsoever, and was the cause of the initial frustration. Once the operational files had been edited appropriately, no problems were encountered as the tests proceeded. A VB 100% therefore wings its way to Finland.


Of note with this product were the relative sizes of the product and definition files. The full product totalled 6.9 MB, a little above the average for the *Linux* products reviewed here. The additional definition files, however, were 7.1 MB in size – larger than the product itself.

## FRISK F-Prot Antivirus 3.16.6

| | | | |
|---|---|---|---|
| **ItW File** | 100.00% | **Macro** | 100.00% |
| **ItW File (o/a)** | 100.00% | **Standard** | 99.82% |
| **Linux** | 100.00% | **Polymorphic** | 100.00% |

Last year *FRISK*'s product was notable for its slow speed of scanning. However, this problem seems to have been banished in the intervening months.


No longer as closely related to *FSAV* as it once was, the two products are beginning to show a divergence in their test results. Not a major divergence though, since *FRISK* missed only one sample across the whole test set (which was not in the wild) and duly qualifies for a VB 100% award.

## Grisoft AVG 7 Anti-Virus 7.0.15

| | | | |
|---|---|---|---|
| **ItW File** | 100.00% | **Macro** | 100.00% |
| **ItW File (o/a)** | 100.00% | **Standard** | 97.15% |
| **Linux** | 41.67% | **Polymorphic** | 83.72% |

*AVG* is yet another *Dazuko*-based scanner, although unlike most other *Dazuko*-based products it does not set up shares to be scanned automatically – these must be designated manually. Even with this requirement, however, less tweaking was required this time than was necessary last year, for which my thanks go to *Grisoft*.


Files missed during scanning were very much the same as those usually missed by *AVG* – the weakness of the scanner lying in complex polymorphic viruses. Since real viruses are almost never seen these days, however, this is not the issue that it once appeared likely to be. With perfect detection of samples in the wild, *AVG* is worthy of a VB 100% award.

## H+BEDV Antivir 2.1.3-17

| | | | |
|---|---|---|---|
| **ItW File** | 100.00% | **Macro** | 100.00% |
| **ItW File (o/a)** | 100.00% | **Standard** | 100.00% |
| **Linux** | 91.67% | **Polymorphic** | 100.00% |

Despite ostensibly being the same product as *Avira*, the *Antivir* package weighs in at well over twice the size of its relative (a sturdy 8.8 MB in comparison with *Avira*'s 3.4 MB). The reason for this soon became apparent, however, since a Java-based GUI is included in the package.

| On-demand tests | ItW file | | Macro | | Polymorphic | | Standard | | Linux | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Number missed | % | Number missed | % | Number missed | % | Number missed | % | Number missed | % |
| Alwil Avast | 0 | 100.00% | 18 | 99.56% | 113 | 93.57% | 15 | 99.36% | 14 | 50.00% |
| Avira Avira | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 2 | 91.67% |
| CAT Quick Heal | 0 | 100.00% | 74 | 98.20% | 314 | 96.25% | 104 | 96.33% | 7 | 60.00% |
| Doctor Web Dr.Web | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% |
| Eset NOD32 | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% |
| F-Secure Anti-Virus | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 1 | 93.33% |
| FRISK F-Prot Antivirus | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 1 | 99.82% | 0 | 100.00% |
| Grisoft AVG | 0 | 100.00% | 0 | 100.00% | 425 | 83.72% | 44 | 97.15% | 17 | 41.67% |
| H+BEDV Antivir | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 2 | 91.67% |
| Kaspersky Anti-Virus | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% |
| MicroWorld eScan | 177 | 58.67% | 0 | 100.00% | 0 | 100.00% | 20 | 97.71% | 7 | 60.00% |
| McAfee LinuxShield | 0 | 100.00% | 0 | 100.00% | 0 | 100.00% | 2 | 99.82% | 0 | 100.00% |
| Norman Virus Control | 0 | 100.00% | 6 | 99.85% | 147 | 92.09% | 6 | 99.66% | 1 | 93.33% |
| SoftWIN BitDefender | 0 | 100.00% | 33 | 99.14% | 6 | 99.73% | 22 | 99.23% | 11 | 53.33% |
| Sophos SWEEP | 0 | 100.00% | 8 | 99.80% | 0 | 100.00% | 15 | 99.30% | 8 | 58.33% |
| Trend Micro ServerProtect | 0 | 100.00% | 0 | 100.00% | 182 | 96.22% | 8 | 99.66% | 4 | 93.33% |
| VirusBuster VirusBuster | 3 | 99.76% | 0 | 100.00% | 3074 | 77.01% | 66 | 97.24% | 29 | 53.33% |

This was not tested. Other than this difference, *Antivir* and *Avira* were identical. Command line options and detection were the same for both products, with timing tests the same within the tolerances of such tests.

It should not take great detective skills, therefore, to realise that *Antivir* also receives a VB 100%.

## Kaspersky Anti-Virus 5.0.3.0 build 15

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

There was a problem with *KAV* concerning the installation of additional virus databases. However, I was forewarned of this by the product developers, and so I was spared some frustration.


In last year's comparative tests the product's documentation and installation in general proved problematic, but there were no issues with these on this occasion, which was something of a relief.

Operating as a vfs object, the Samba scanning operated perfectly, blocking all infected objects. The on-demand scanner equalled this detection, with a VB 100% for *Kaspersky* as the result.

## MicroWorld eScan Antivirus 1.0A

| | | | |
|---|---|---|---|
| **ItW File** | 58.67% | **Macro** | 100.00% |
| **ItW File (o/a)** | 100.00% | **Standard** | 97.71% |
| **Linux** | 60.00% | **Polymorphic** | 100.00% |

A new addition to the *Linux* comparative, *eScan* proved to be a mixed bag of problems and delights. Of all the products supplied using *Dazuko*, *eScan* is the only one that includes the *Dazuko* source and that configures and makes *Dazuko* automatically during installation. After this pleasant surprise the GUI was launched, which is the interface for on-access scanning, and it was here that matters became a little confusing, since the GUI offers no obvious way in which to perform on-demand scans.

After updating the product the GUI indicated new definition dates and thus testing was commenced. The results were very good indeed on access. On demand was another matter however – a whole host of files were missed. These were a mixture of older and newer files, though most were newer. Assuming this to be a definitions issue the updates were checked again, but all seemed to be in order.

Another oddity was encountered upon invoking the on-demand scanner on a directory with no leading or trailing '/' supplied. Here, a segmentation fault was triggered. With these problems on demand it comes as no surprise that a VB 100% cannot be awarded to *eScan* on this occasion.

## McAfee LinuxShield 1.1.0.665.i686

| | | | |
|---|---|---|---|
| **ItW File** | 100.00% | **Macro** | 100.00% |
| **ItW File (o/a)** | 100.00% | **Standard** | 99.82% |
| **Linux** | 100.00% | **Polymorphic** | 100.00% |

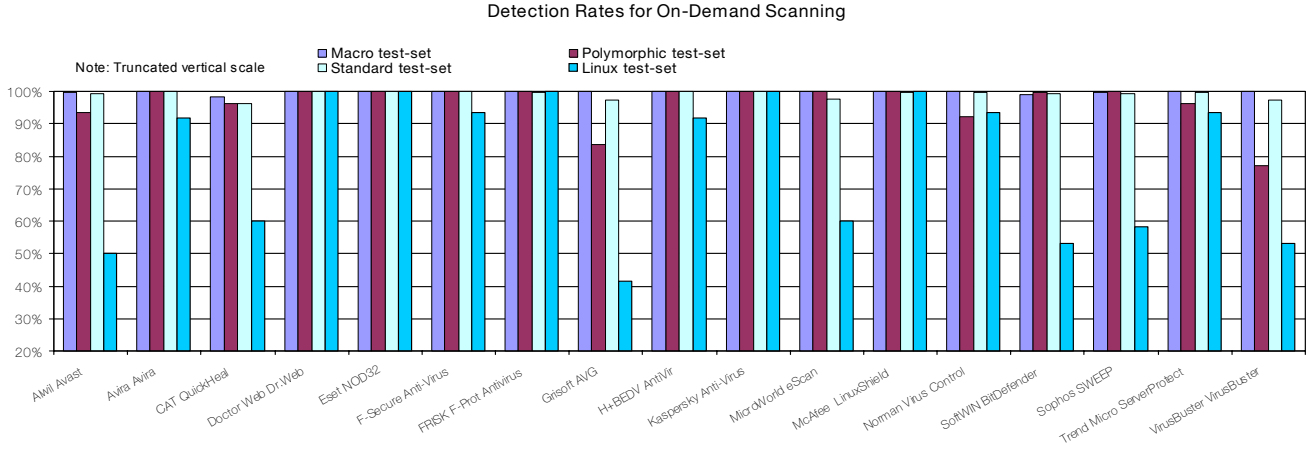
*LinuxShield* is the new name for *McAfee*'s *Linux* offering. In this case a GUI is mandatory. The only way to perform on-demand scans conveniently is through the GUI. Performing them from the command line requires scan parameters to be set up via the GUI – so in this case the GUI was used for on-demand testing. Updating seemed a little awkward from a local directory, in that engine updates worked, while definition updates did not. These were performed by copying the definitions manually to the correct area.

The only false positive of the tests occurred with *LinuxShield*, though this was not a serious one – the file being flagged as a 'program' rather than as a real virus. Since the file in question was a reboot utility, this flag seemed justified. Scanning was good as far as detection was concerned, so this new incarnation gains a VB 100% where its predecessor failed.

## Norman Virus Control 5.70.01

| | | | |
|---|---|---|---|
| **ItW File** | 100.00% | **Macro** | 99.85% |
| **ItW File (o/a)** | 100.00% | **Standard** | 99.66% |
| **Linux** | 93.33% | **Polymorphic** | 92.09% |

The on-access functionality in *Norman Virus Control* (*NVC*) is new – in fact, it is so new that some documentation states that it does not yet exist. The on-access scanner uses *Dazuko* to scan and performed well. However, it seems that it can be configured only via the Java-based GUI. On-demand scanning, meanwhile, is perfectly configurable through the command line. In the end, results for *NVC* were much the same as have been seen in recent comparatives on other platforms and *NVC* is awarded a VB 100%.



| Hard Disk Scan Rate | Executables | | | OLE Files | | | Zipped Executables | | Zipped OLE Files | | Linux 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) | Time (s) | Throughput (kB/s) |
| Alwil Avast | 138 | 3963.3 | | 12.1 | 6556.5 | | 23 | 6931.2 | 6.1 | 12230.7 | 6.0 | 4503.4 |
| Avira Avira | 416 | 1314.7 | | 7.2 | 11018.6 | | 193 | 826.0 | 12.4 | 6016.7 | 4.3 | 6283.9 |
| CAT Quick Heal | 64 | 8545.8 | | 13.0 | 6102.6 | | 45 | 3542.6 | 17.3 | 4312.6 | 4.4 | 6141.0 |
| Doctor Web Dr.Web | 186 | 2940.5 | | 11.6 | 6839.1 | | 85 | 1875.5 | 15.3 | 4876.3 | 5.4 | 5003.8 |
| Eset NOD32 | 40 | 13673.3 | | 4.5 | 17629.7 | | 19 | 8390.3 | 1.5 | 49738.3 | 2.2 | 12282.1 |
| F-Secure Anti-Virus | 168 | 3255.5 | | 15.9 | 4989.5 | | 86 | 1853.7 | 32.7 | 2281.6 | 8.0 | 3377.6 |
| FRISK F-Prot Antivirus | 114 | 4797.7 | | 4.8 | 16527.9 | | 50 | 3188.3 | 5.3 | 14076.9 | 2.0 | 13510.3 |
| Grisoft AVG | 99 | 5524.6 | | 11.2 | 7083.4 | | 74 | 2154.3 | 13.4 | 5567.7 | 16.3 | 1657.7 |
| H+BEDV Antivir | 358 | 1527.7 | | 7.7 | 10303.1 | | 201 | 793.1 | 11.0 | 6782.5 | 4.9 | 5514.4 |
| Kaspersky Anti-Virus | 143 | 3824.7 | | 15.1 | 5253.9 | | 62 | 2571.2 | 16.9 | 4414.6 | 8.3 | 3255.5 |
| MicroWorld eScan | 66 | 8286.9 | | 16.1 | 4927.6 | | 80 | 1992.7 | 46.2 | 1614.9 | 8.0 | 3377.6 |
| McAfee LinuxShield | 170 | 3217.2 | [1] | 12.0 | 6611.1 | | 79 | 2017.9 | 17.0 | 4388.7 | 7.0 | 3860.1 |
| Norman Virus Control | 523 | 1045.8 | | 8.1 | 9794.3 | | 481 | 331.4 | 8.2 | 9098.5 | 1.7 | 15894.4 |
| SoftWIN BitDefender | 304 | 1799.1 | | 7.5 | 10577.8 | | 165 | 966.2 | 7.9 | 9444.0 | 15.9 | 1699.4 |
| Sophos SWEEP | 64 | 8545.8 | | 11.4 | 6959.1 | | 45 | 3542.6 | 12.7 | 5874.6 | 5.9 | 4579.8 |
| Trend Micro ServerProtect | 88 | 6215.1 | | 5.0 | 15866.8 | | 29 | 5497.1 | 7.0 | 10658.2 | 4.0 | 6755.1 |
| VirusBuster VirusBuster | 137 | 3992.2 | | 9.7 | 8178.7 | | 83 | 1920.7 | 15.0 | 4973.8 | 6.7 | 4032.9 |

## SoftWIN BitDefender 1.6.2-0

| | | | |
|---|---|---|---|
| **ItW File** | 100.00% | **Macro** | 99.14% |
| **ItW File (o/a)** | 100.00% | **Standard** | 99.23% |
| **Linux** | 53.33% | **Polymorphic** | 99.73% |

*BitDefender*'s performance in the last comparative review was marred both by unexpected missed files and by a tendency for the Samba share to lose connections. Happily, both of these problems have been fixed in this latest version.

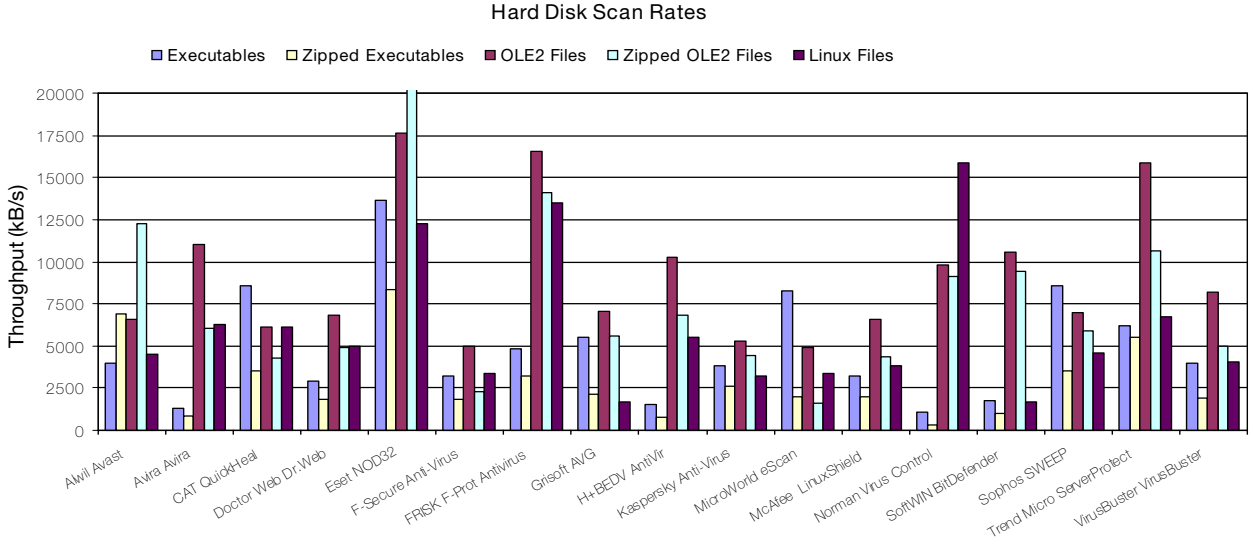

The only slight surprise for *BitDefender* was the fact that the product missed more files on demand than on access. Detection results were generally good, however, and no files were missed in the all-important ItW test set. These improvements are sufficient to justify the award of a VB 100% for this test.

## Sophos SWEEP 3.91.0

| | | | |
|---|---|---|---|
| **ItW File** | 100.00% | **Macro** | 99.80% |
| **ItW File (o/a)** | N/A | **Standard** | 99.30% |
| **Linux** | 58.33% | **Polymorphic** | 100.00% |

As mentioned already, *Sophos*'s *SWEEP* is the only product in this comparative review which exists purely as an on-demand scanner. Its lack of an on-access scanner discounts it instantly from a VB 100% award.

Other than this, results for on-demand scanning were good, although detection rates were slightly low in the *Linux* test set. However, the misses in this set are indicative of a general issue with some of the *Linux* worms in the test set. Some of these, such as Linux/Lion, are packaged as archives in their transmitted state. Along with several other products, *SWEEP* does not detect inside archives in its default state.




## Trend Micro ServerProtect 2.452.00 7.510

| | | | |
|---|---|---|---|
| **ItW File** | 100.00% | **Macro** | 100.00% |
| **ItW File (o/a)** | 100.00% | **Standard** | 99.66% |
| **Linux** | 93.33% | **Polymorphic** | 96.22% |

*ServerProtect* was among the first products to operate within a GUI and continues to do so – with scanning outside the GUI not easy. For this reason tests were performed from within the GUI. Polymorphic samples represented the bulk of misses for *Trend*'s product, with there being a noticeable increase in the number of misses when scanning on access. These notwithstanding, results were ample for a VB 100% to be awarded. A slight worry was the continuation of a bug that was noted in the product a year ago. When accessing the http-based GUI one URL is slightly garbled. This occurs in exactly the same way today as it did 12 months ago.


## VirusBuster VirusBuster 2005 1.1.1

| | | | |
|---|---|---|---|
| **ItW File** | 99.76% | **Macro** | 100.00% |
| **ItW File (o/a)** | 99.76% | **Standard** | 97.24% |
| **Linux** | 53.33% | **Polymorphic** | 77.01% |

*VirusBuster*'s scanner showed some strange patterns in detection and as a result tests were performed in several fashions. During the course of these it became apparent that files can neither be either deleted nor quarantined if they contain infected objects such as *PowerPoint* objects. However, the main reason for the additional scans was the fact that samples of W32/Bugbear.B were missed both on access and on demand. Since these are in the wild, this was enough to deny *VirusBuster* a VB 100% award.

## CONCLUSION

As was hoped at the outset of this review, the stability of the scanners in this test has shown significant improvement since last year and, in many cases, the installation procedures have become substantially simpler.

However, there were still some issues with updating products and some products are still far less than intuitive to set up. The arrival of more GUIs on the scene is something of a mixed blessing. On the one hand the use of a GUI can be easier for configuration – but on the other, a *Linux* application without full command-line control seems inherently wrong. Although I foresee that the scanners will become increasingly similar to *Windows* applications as far as GUI-centric operation is concerned, it would be appreciated if this were also extended to general ease of use.

**Technical details**

**Test environment:** Identical 1.6 GHz Intel Pentium machines with 512 MB RAM, 20 GB dual hard disks, DVD/CD-ROM and 3.5-inch floppy drive running Red Hat Linux 9, kernel build 2.4.20-8 and Samba version 2.2.7a. An additional machine running Windows NT 4 SP 6 was used to perform read operations on the Samba shared files during on-access testing.

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

**HITBSecConf 2005 'deep knowledge E-security conference' takes place 10–13 April 2005 in Bahrain**. For full details see http://www.hitbsecconf.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**. This year the conference theme is 'Technical, legal and social aspects of IT security'. For full details and online registration 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/.

**AusCERT 2005 takes place 22–26 May 2005 in Gold Coast, Australia**. Programme details and online registration are available at http://conference.auscert.org.au/.

**The third International Workshop on Security in Information Systems, WOSIS-2005, will be held 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 programme 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**. The deadline for submitting paper proposals is 1 May 2005; registration for the event is now open. 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 Network Security Conference takes place 19–21 September 2005 in Las Vegas, NV, USA**. The conference is designed to meet the education and training needs of the seasoned IS professional as well as the newcomer. For details see http://www.isaca.org/.

**The 15th Virus Bulletin International Conference, VB2005, will take place 5–7 October 2005 in Dublin, Ireland**. The conference programme can be found on the *VB* website. For more information or to register online see http://www.virusbtn.com/.

**RSA Europe 2005 will be held 17–19 October 2005 in Vienna, Austria**. For more details see http://www.rsaconference.com/.

**WORM 2005 (the 3rd Workshop on Rapid Malcode) will take place 11 November 2005 in Fairfax, VA, USA**. The workshop will provide a forum to bring together ideas, understanding and experiences bearing on the worm problem from a wide range of communities, including academia, industry and the government. The organisers are currently seeking submissions from those wishing to present at the workshop. Full details can be found at http://www1.cs.columbia.edu/~angelos/worm05/.




## 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

## SPAMMER INVESTIGATED IN UKRAINE

A spammer is undergoing criminal investigation in the Donetsk region of the Ukraine. This is the first case of spamming to be pursued by the country's law enforcement authorities following amendments to the Criminal Procedures Code which came into force earlier this year. The investigation began after a representative of a local ISP lodged a complaint with the police department responsible for economic crimes, saying that the ISP's network was being flooded with spam. If found guilty, the spammer faces a fine of $1600–$3200 or up to three years imprisonment.

## SPAMMER VS SPAMMED

A US man is being sued for accusing a company of sending him spam. Self-proclaimed 'anti-spammer' Mark Mumma says that, after receiving four unsolicited emails from online travel agent *Cruise.com*, he decided to lodge a complaint with the travel agent's parent company *Omega World Travel*. After lodging his complaint Mumma claims he was given the impression that his email address would be removed from the company's circulation list – however, he continued to receive emails from *Cruise.com*. Mumma documented this fact, along with the history of the complaint, on his website 'sueaspammer.com'.

Unfortunately, *Omega World Travel* took umbrage at the fact that Mumma had exposed the company for alleged spamming activities and he now faces a lawsuit.

*Omega* argues that Mumma violated their trademark and copyright by using images of the company's founders and the company's logo on his website, and allege that Mumma defamed individuals associated with *Cruise.com* by posting personal insults on his site. Mumma has filed a motion to dismiss the case based on jurisdiction, but at the time of writing no decision has been made.

This is not the first case of an illeged spammer taking legal action against the spammed this year. In January New Hampshire firm *Atriks* filed a lawsuit against Jay Stuler who, it alleges, caused financial harm to the firm by reporting its spamming actions to his ISP.

In other courtroom news, a US judge has overturned a conviction in one of last year's high profile anti-spam cases. Judge Thomas D. Horne ruled last month that there was insufficient evidence for the conviction of Jessica DeGroot alongside her brother, prolific spammer Jeremy Jaynes (aka Gavin Stubberfield). In November 2004 a jury convicted DeGroot and Jaynes on several felony counts of using fraudulent means to send unsolicited bulk email. DeGroot was fined $7,500, while Jaynes was sentenced to nine years imprisonment. However, Judge Horne dismissed DeGroot's conviction, ruling that it had been made without 'rational basis'. Jaynes's conviction was upheld.

## EVENTS

CEAS 2005, the Second Conference on Email and Anti-Spam, will be held 21–22 July 2005 at Stanford University, CA, USA. For more information see http://www.ceas.cc/.

INBOX IT takes place 1–2 June 2005 in San Jose, CA, USA. The event will focus on all aspects of email including spam, phishing, zombies, outbound controls, encryption and the latest in new security technologies and techniques. More information is available at http://www.inboxevent.com/.

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

# FEATURE

## BAYESIAN NOISE REDUCTION

*Jonathan Zdziarski*
DSPAM, USA

Any reputable modern spam filter can deliver 99 per cent accuracy or beyond, and most users are quite content with this. However, filter authors are always looking for new ways to improve their results, and I suspect this is so because statistical filtering runs along the same lines as NASCAR: all the cars (filters) are fast, but it's squeezing the last 5mph out of them that makes the science exciting. While the best-of-breed spam filters available today are well-oiled machines, I think there is room for improvement. My biggest complaint about present-day filters is their lack of true lexical intelligence. When we think of spam filtering, most of us think along the lines of content-based filtering. Today's generation of statistical filters are wired simply to scan through emails and look for the most interesting 'buzzwords' by which to judge their disposition.

However, spam filters face an obstacle: bogus data. Spammers seem to have grasped the idea of how typical content-based filters work, and at the very least, they know that, to improve the chances of their emails being allowed through, they need to hide the 'bad' words and add some 'good' words. Spammers now inject anything into their emails, from nonsense text to a target group's web page. Most of the time, user data is far too specialized for filters to crack on words taken from obscure books or website lingo that we've never used. Once in a while, however, spammers get lucky, including just the right text in the right quantity to trigger a response that is uncertain enough for the message to be passed into our inbox. More concerning is the prospect that spammers could find ways to mine even more specific data from users through the use of Internet worms, web bugs, or similar means.

If we are ever going to break 'five 9s' accuracy, I believe content-based filters need to evolve into concept-based filters. Many of the technologies I have developed into my own project (DSPAM) have been designed with this goal in mind. The idea is to recognise not only individual words (or even word pairs), but also concepts (e.g. 'free porn'), and then classify based on what falls into that concept.

Bayesian Noise Reduction (BNR) is one of the technologies I have designed in an attempt to give the filter a 'lexical brain' – an intelligence that allows the filter to look at language in a similar way to humans. In order to form concepts, spam filters must have something they do not presently have: a context.

Think of BNR as a way to identify words that are out of context: if we are having a conversation and I say something out of context, there is a good chance that the words I said would have had a completely different meaning if they had been in context. This problem flows into language classification where a token can resolve to one disposition when in context (e.g. the word 'free'), but have a completely opposite disposition when out of context (such as in a list of nonsense text injected by a spammer). Today's filters have no way of dealing with out-of-context words or phrases because they have no idea of context.

BNR identifies out-of-context data by creating its own contexts. It creates a series of machine-generated contexts around a sample of text (the message body), and then identifies data that contradicts itself within the context it has created. The process is illustrated below using three basic steps that any statistical filter should be able to implement.

### INSTANTIATION PHASE

Let's take a look at some text your filter might happen across while reading your email:

```
Mom Would Be Proud Try Viagra Now!
```

When your statistical filter reads 'Mom Would Be Proud. Try Viagra Now!', it will assign a series of probabilities (values) to each word (because that's what filters do).

| Text: | Mom | Would | Be | Proud | Try | Viagra | Now! |
|---|---|---|---|---|---|---|---|
| Values: | 0.60 | 0.34 | 0.71 | 0.20 | 0.91 | 0.99 | 0.99 |

The first part of the noise reduction process involves instantiating a series of artificial contexts, or patterns, around this text. The first step is to pigeonhole each of the values assigned by the filter into a band, rounded to the nearest 0.05. This helps to limit the total number of patterns we are likely to come up with.

| Text: | Mom | Would | Be | Proud | Try | Viagra | Now! |
|---|---|---|---|---|---|---|---|
| Values: | 0.60 | 0.34 | 0.71 | 0.20 | 0.91 | 0.99 | 0.99 |
| Bands: | 0.60 | 0.35 | 0.70 | 0.20 | 0.90 | 1.00 | 1.00 |

Next, we simply chain the bands together, three by three, to create patterns:

```
0.60_0.35_0.70   0.35_0.70_0.20   0.70_0.20_0.90
0.20_0.90_1.00   0.90_1.00_1.00
```

Each pattern represents the bands for three adjacent tokens in our sample text. We instantiate patterns for the entire body of our message, which leaves us with a series of artificial contexts.

### TRAINING PHASE

Once we have a series of artificial contexts instantiated for an email, we need to spend time learning them in a very similar fashion to the rest of the tokens in our database. Each token is given a spam counter and a nonspam counter,

and we calculate a probability for each pattern. I use Paul Graham's approach to assigning token values:

P = (spamHits / totalSpam) / (spamHits / totalSpam + innocentHits / totalInnocent)
*[No bias is used when calculating pattern values]*

| | |
|---|---|
| bnr.s.1.00.0.00.0.45 | [0.91111] |
| bnr.c.0.25.1.00.1.00 | [0.99990] |
| bnr.s.0.35.1.00.1.00 | [0.99990] |
| bnr.s.1.00.1.00.0.20 | [0.99990] |
| bnr.s.1.00.1.00.0.25 | [0.99990] |
| bnr.s.0.55.1.00.1.00 | [0.99990] |
| bnr.c.1.00.1.00.0.35 | [0.99990] |
| bnr.s.0.25.1.00.1.00 | [0.99990] |
| bnr.c.1.00.1.00.0.15 | [0.99990] |
| bnr.c.0.15.1.00.1.00 | [0.99990] |
| bnr.c.0.10.1.00.1.00 | [0.99990] |
| bnr.s.0.35.1.00.0.40 | [0.99990] |
| bnr.s.0.40.0.35.1.00 | [0.99990] |
| bnr.c.0.20.1.00.1.00 | [0.99990] |
| bnr.s.0.00.0.00.0.45 | [0.99990] |

*Table 1: Learned pattern contexts.*

After a handful of email has been processed in this fashion, our contexts will take on a disposition just like any other token, as illustrated in Table 1. Some contexts will have a very innocent or very guilty disposition, and others will be less interesting. We want to identify contexts that are both very extreme in their value and self-contradictory. A pattern context must meet two basic criteria to be interesting enough to use:

1. The pattern's value must exceed an exclusionary radius of 0.25 from neutral, or ABS(0.5-P). For a typical Bayesian filter, this means that the pattern's value must resolve to 0.00–0.25 or 0.75–1.00.
2. The pattern must hold at least one data point with a value at least 0.30 distant from the pattern's value, or ABS($P_P$-$P_W$) where $P_P$ is the value of the pattern and $P_W$ is the value of the word, or token.

In Table 1, we see that the pattern 1.00_0.00_0.45 has an extremely guilty value (a 91 per cent likelihood of being spam). Not only is this very interesting, but the fact that the pattern includes an extremely innocent token (0.00) is a good indication that this is a token we want to examine.

## DUBBING PHASE

Now let's take a logical look at what we have accomplished. Given the pattern 1.00_0.00_0.45, which our filter trained to a value of 91 per cent, our filter has discovered that the presence of an extremely guilty token (1.00) next to an extremely innocent token (0.00), next to a token we have not seen before (0.45 is the neutral value I assign to new tokens), is guilty. That is, this pattern of token values (regardless of the actual words used) is guilty.

If this pattern is guilty, then we must reach the logical conclusion that the token which the filter previously learned as 0.00 is contradictory in its present context – i.e. it must be *out of its normal context*.

The dubbing phase is quite simply the omission of these anomalies. Given:

```
bnr.s.0.35.0.05.0.80   [0.99990]
bnr.s.0.05.0.80.1.00   [0.99990]

Text:    Your   Terminal   TRY    VIAGRA!
Values:  0.34   0.04       0.81   0.99
Band:    0.35   0.05       0.80   1.00
```

We then dub out the inconsistencies – i.e. any token in the pattern whose value is further than 0.30 from the pattern's value. So instead of seeing 'Your Terminal TRY VIAGRA!', we now see:

```
Text:    Blah   Blah  TRY   VIAGRA!
Values:  -.-    -.-   0.81  0.99
```

Or we could get even more creative and change the polarity of the out-of-context tokens to match that of the context, which provides a bit of moral satisfaction (and possibly a more accurate result):

```
Text:    Your   Terminal   TRY    VIAGRA!
Values:  0.99   0.99       0.81   0.99
```

## END RESULT

The end result after processing messages against this algorithm is quite impressive. When legitimate messages are processed we find a significant reduction in the number of guilty identifiers that could lead to a false positive. After processing many spams there is a significant reduction in the number of innocent identifiers that could lead to a spam misclassification – and this all takes place without the noise reduction algorithm having any knowledge about the true disposition of the message.

After performing tests on random system users, I found that the BNR algorithm improved confidence by an average of 20 per cent and in a few isolated cases (which I call false

| | | |
|---|---|---|
| Total samples analyzed | 3948 | 2280 |
| Total improved confidence | 2523 | 1522 |
| Total decreased confidence | 26 | 16 |
| Total N/C in confidence | 1399 | 742 |
| Avg increase in confidence | 21.51% | 20.80% |
| Avg decrease in confidence | 5.26% | 4.00% |

*Confidence calculated using Robinson's Geometric Mean Test Inverted.*

*Table 2: Bayesian Noise Reduction (BNR) illustrating improved confidence in most samples.*

positives), only reduced confidence by about 5 per cent. Table 2 shows the results for two of these users on my system.

## EFFICACY

I have found that BNR's effectiveness (and long-term efficacy) depends on how its pattern contexts are trained by the user's filters. At present, I am training on every message processed (and re-training on errors), but I have found that training only on hard-to-classify messages makes BNR a little more sensitive to different types of noise. There is also a threshold for purging which should be developed through trial and error. Dividing all of the counter totals by two at certain milestones might help keep the pattern contexts sufficiently dynamic to adapt to new types of context. Training philosophy will affect BNR's performance, and so it is a good idea to find a happy medium through testing.

Since BNR behaves based on the context values it has learned for a specific user, actual mileage may vary. I am confident, however, that this approach will come in handy as spammers continue to grow their word-mining databases. At some point, spammers will be able to generate enough accurate junk to increase their success rate against typical content-based statistical filters. The great thing about this algorithm is that its function is abstracted from the actual words. In order to circumvent this type of approach, a spammer not only needs to mine words that are likely to be 'innocent', but they also need to mine both guilty and neutral words to the nearest 0.05, as well as the learned patterns and values from a user's filter, and then put it all together to create a series of artificially 'in-context' junk text. This, at the very least, is computationally infeasible today.

## FINAL THOUGHTS

The BNR algorithm appears to be very useful at identifying out-of-context data within any type of message (good or spam), and does its job remarkably well. During the summer, I tested this algorithm against my own corpus of mail and was very surprised to see accuracy jump from 99.96 per cent to an astonishing 99.985 per cent (from 1 error in 2,500 to 1 error in 7,500).

Bayesian Noise Reduction has been implemented in DSPAM version 3.4 and above (older versions sported a more heuristic approach), and is available in a GPL library for other filter authors at http://bnr.nuclearelephant.com/ (where a related white paper and MIT Spam Conference presentation can also be downloaded). Open source filter authors are invited to grab the library and give the implementation a spin.

# SUMMARY

## ASRG SUMMARY: MARCH 2005

*Helen Martin*

Devdas Bhagat opened this month's dialogue with a question about trust and trust propagation. He pointed out that most trust mechanisms used to identify legitimate mail try to push trust information based on the claimed SMTP sender/sender domain – but these fail because, in general, they more or less trust what the sender claims to be. Devdas asserted that the one really trustworthy piece of information in the entire SMTP transaction is the IP address of the peer, and as such, basing trust on the administrator of the peer IP address would be more useful than basing it on the domain of the sender or on the address itself.

Devdas has drafted a proposal for a DNSBL which would allow multiple sites to communicate their trust of different IP addresses, and allow site administrators to define trust levels for other domains (his full proposal can be read at http://nixcartel.org/~devdas/multisystem-protocol-proposal.txt). Peter J. Holzer agreed that we should be looking at trusting the sending hosts rather than the sending domains/addresses, but that this might change as spammers move from direct-to-mx sending to using the smarthost of zombies.

Daniel Feenberg took it upon himself to post some 'actual research' to the list – a relatively rare occurrence in the short history of the ASRG. Daniel's research took the form of a quantitative comparison of 16 well-known DNSBLs, from which he concluded that he was happy to stand by his conviction that DNSBLs are the long-term best approach to spam suppression.

Peter Kay reported that, after nearly six years and 'lots of hand wringing', he and his colleagues at *Titan Key* had been awarded a US patent for their user-level blacklisting as was first disclosed to the ASRG in 2003 (see http://www.shaftek.org/publications/asrg-ipr.html#4.2). This news sparked a torrent of emails questioning the soundness of the patent (specifically *Titan*'s assertion that user-level black/whitelisting was non-existent before January 2000) – and, indeed, questioning the value of patents in general.

It was a somewhat brave move, in light of the general mood of the group, for Phillip Hallam-Baker to draw members' attention to his own patent application. Phillip's key claim is a means of accrediting end users that does not require identity accreditation, and met with a substantially less hostile reaction.

*A more detailed explanation of Phillip's patent filing can be found in the ASRG archives at http://www1.ietf.org/mail-archive/web/asrg/current/, along with the rest of this month's discussions.*