Fighting malware and spam

CONTENTS

2  COMMENT
   Public liability insurance for computer intrusion

3  NEWS
   Festive greetings
   VB2009 Geneva: call for papers
   Apple urges Mac users to install AV

3  VIRUS PREVALENCE TABLE

4  TECHNICAL FEATURE
   Anti-unpacker tricks – part one

9  OPINION
   Repercussions of dynamic testing

12 SPOTLIGHT
   Frame4: in the picture

14 COMPARATIVE REVIEW
   Windows Vista x64

26 END NOTES & NEWS

IN THIS ISSUE

TRICKS OF THE TRADE
Unpackers have been around for as long as packers themselves, but anti-unpacking tricks are a more recent development and have grown quickly both in number and, in some cases, complexity. Peter Ferrie describes a range of anti-unpacking tricks.

DYNAMIC WARNING
Roel Schouwenberg has a word of caution regarding the attention currently being devoted to dynamic testing of anti-malware products.

VB100: WINDOWS VISTA X64
The final VB100 of the year sees a double whammy of potential pitfalls for comparative participants – the Vista operating system as well as the x64 architecture. John Hawes reveals how the products coped.

vbSpam supplement
This month: anti-spam news and events, and Alexandru Cosoi describes a method that attempts to deal with the phishing problem at the browser level, combining both whitelisting and content-based solutions in a web page forgery detector.
PUBLIC LIABILITY INSURANCE FOR COMPUTER INTRUSION

Spam and malware are problems for everyone who uses the Internet, and common methods that are used to combat the phenomena – such as filtering the junk and controlling access – do not seem to be much of a deterrent for the attackers.

The creation and distribution of malware and the sending of spam are activities that are driven by profit, and they will continue for as long as the benefits to the perpetrators exceed their cost. But these activities also impose costs on other users of the Internet: spillover costs. In any activity, spillover costs are a sign that the activity has exceeded an acceptable level. There must be a way to counter the spillover costs by diminishing the benefits or increasing the penalties for the perpetrators.

The most obvious solution is to increase the penalties for spamming and unauthorized computer intrusion – and many countries now have extensive anti-spam and computer crime laws, but they have had little impact on the levels of these crimes. Suggestions for economic solutions, such as imposing a minimal price for each email sent, have also had little success. So far, one thing no one seems to have considered is the idea of tackling the other categories of users – those who purchase the products/services advertised in spam, and those who leave their computers unprotected and consequently get infected.

Tracking down those who make purchases from spam is likely to be very difficult – which leaves us with those who do not secure their PCs.

In order to understand how a greater number of protected computers would be beneficial, let’s look at motor insurance. Uninsured car drivers cause higher insurance premiums (because if an uninsured driver causes an accident and cannot pay the damage, the other driver(s) have to collect from their own insurance companies, driving their premiums upwards). Thus driving an uninsured car imposes spillover costs on all the people you meet on the road. However, the higher the insurance premiums, the less likely that drivers will take out insurance. There is no way to get out of that vicious circle without help from the outside – which comes in the form of mandatory insurance. Mandatory motor insurance brings down the cost of insurance (spillover cost) both because there are fewer uninsured drivers to drive up premiums, and because the more people buy insurance the more likely it is to be offered at a lower cost.

What would happen if the use of security solutions was mandatory? More people would install security products, which would have multiple effects. First, with more machines protected it would be harder for botnet masters to recruit new zombie machines, thus increasing their costs, which in turn would increase the cost of spamming and decrease its profitability. It would also increase the revenues of security companies which, in a highly competitive market, could lead to an overall decrease in the cost of the security products themselves. That would complete the circle, with the lower cost of solutions combined with their mandatory use resulting in a larger number of people protecting their computers.

The key to all this is that by installing a security suite you not only protect yourself, but you increase the safety of the whole community as you protect the rest of us from the menace you would become once infected. Thus it might be viewed as a form of liability insurance.

This approach does face significant obstacles – such as the fact that legislation would have to be passed, which would take time. Furthermore, making computers harder to attack in one country would have little effect unless other countries took action as well – otherwise the attackers would simply shift the focus of their operations to another geographical area. Complications would also arise regarding enforcement of the legislation. A possible solution would be to insist that every buyer has a licence for a security solution when buying a new computer or any major computer component such as the motherboard.

It is my belief that making the use of security products mandatory could make the lives of spammers and other online criminals so much more difficult that it would act as a deterrent and make the Internet a safer place for all.
FESTIVE GREETINGS

The members of the VB team extend their warm wishes to all Virus Bulletin readers for a very happy holiday season and a healthy, peaceful and prosperous new year.

VB2009 GENEVA: CALL FOR PAPERS

Virus Bulletin is seeking submissions from those wishing to present papers at VB2009, which will take place 23–25 September 2009 at the Crowne Plaza, Geneva, Switzerland.

The conference will include a programme of 40-minute presentations running in two concurrent streams: Technical and Corporate. Submissions are invited on all subjects relevant to anti-malware and anti-spam. In particular, VB welcomes the submission of papers that will provide delegates with ideas, advice and/or practical techniques, and encourages presentations that include practical demonstrations of techniques or new technologies.

The deadline for submission of proposals is Friday 6 March 2009. For full details of how to submit a paper, along with a list of topics suggested by attendees of VB2008, please see http://www.virusbtn.com/conference/vb2009/call/.

In addition to the 40-minute presentations, a portion of the technical stream will be set aside for 30-minute, ‘last-minute’ technical presentations, proposals for which need not be submitted until three weeks before the start of the conference. Presenting a full paper will not preclude an individual from being selected to present a last-minute presentation. Further details will be released in due course.

APPLE URGES MAC USERS TO INSTALL AV

Battling the common conception among its users that the Mac platform is safe from malware, Apple issued a quiet announcement last month encouraging the ‘widespread use of multiple anti-virus utilities’ on its products.

With increasing numbers of data-stealing trojans and fake anti-malware programs targeting Mac users, the announcement will come as a wake-up call to many who misguidedly consider their preferred platform to be unaffected by the dangers that Windows users deal with on a daily basis. Apple recommended products from McAfee, Symantec and Mac specialist Intego in its announcement.

Prevalence Table – October 2008

<table>
<thead>
<tr>
<th>Malware</th>
<th>Type</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agent</td>
<td>Trojan</td>
<td>22.10%</td>
</tr>
<tr>
<td>Mytob</td>
<td>Worm</td>
<td>17.74%</td>
</tr>
<tr>
<td>Invoice</td>
<td>Trojan</td>
<td>13.06%</td>
</tr>
<tr>
<td>NetSky</td>
<td>Worm</td>
<td>9.41%</td>
</tr>
<tr>
<td>Suspect packers</td>
<td>Misc</td>
<td>6.28%</td>
</tr>
<tr>
<td>Goldun</td>
<td>Trojan</td>
<td>5.66%</td>
</tr>
<tr>
<td>Autorun</td>
<td>Worm</td>
<td>5.15%</td>
</tr>
<tr>
<td>Bagle</td>
<td>Worm</td>
<td>4.24%</td>
</tr>
<tr>
<td>Mydooom</td>
<td>Worm</td>
<td>3.38%</td>
</tr>
<tr>
<td>Mywife/Nyxem</td>
<td>Worm</td>
<td>2.86%</td>
</tr>
<tr>
<td>Zafi</td>
<td>Worm</td>
<td>1.82%</td>
</tr>
<tr>
<td>Downloader-misc</td>
<td>Trojan</td>
<td>1.68%</td>
</tr>
<tr>
<td>Bifrose/Pakes</td>
<td>Trojan</td>
<td>1.21%</td>
</tr>
<tr>
<td>Monder</td>
<td>Trojan</td>
<td>0.96%</td>
</tr>
<tr>
<td>Parite</td>
<td>Worm</td>
<td>0.50%</td>
</tr>
<tr>
<td>FunLove/Ficss</td>
<td>Worm</td>
<td>0.50%</td>
</tr>
<tr>
<td>Virut</td>
<td>Virus</td>
<td>0.48%</td>
</tr>
<tr>
<td>Klez</td>
<td>Worm</td>
<td>0.43%</td>
</tr>
<tr>
<td>Small</td>
<td>Trojan</td>
<td>0.43%</td>
</tr>
<tr>
<td>Sality</td>
<td>Virus</td>
<td>0.32%</td>
</tr>
<tr>
<td>LovGate</td>
<td>Wrom</td>
<td>0.30%</td>
</tr>
<tr>
<td>Ircbot</td>
<td>Wrom</td>
<td>0.30%</td>
</tr>
<tr>
<td>Iframe</td>
<td>Exploit</td>
<td>0.22%</td>
</tr>
<tr>
<td>Cutwail/Pandex/Pushdo</td>
<td>Trojan</td>
<td>0.18%</td>
</tr>
<tr>
<td>Inject</td>
<td>Trojan</td>
<td>0.13%</td>
</tr>
<tr>
<td>Basine</td>
<td>Trojan</td>
<td>0.11%</td>
</tr>
<tr>
<td>Redlof</td>
<td>Worm</td>
<td>0.10%</td>
</tr>
<tr>
<td>Heuristic/generic</td>
<td>Misc</td>
<td>0.09%</td>
</tr>
<tr>
<td>Womble</td>
<td>Wrom</td>
<td>0.08%</td>
</tr>
<tr>
<td>Banload</td>
<td>Trojan</td>
<td>0.08%</td>
</tr>
<tr>
<td>Bagz</td>
<td>Wrom</td>
<td>0.06%</td>
</tr>
<tr>
<td>Mabutu</td>
<td>Wrom</td>
<td>0.02%</td>
</tr>
<tr>
<td>Bugbear</td>
<td>Wrom</td>
<td>0.02%</td>
</tr>
<tr>
<td>Others[1]</td>
<td></td>
<td>0.10%</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>100.00%</td>
</tr>
</tbody>
</table>

[1] Readers are reminded that a complete listing is posted at http://www.virusbtn.com/Prevalence/.
ANTI-UNPACKER TRICKS – PART ONE
Peter Ferrie
Microsoft, USA

Unpackers have been around for as long as packers themselves, but anti-unpacking tricks are a more recent development. Anti-unpacking tricks have grown quickly both in number and, in some cases, complexity. This paper is an addendum to a paper presented at the CARO workshop in May this year [1], and describes some of the anti-unpacking tricks that have come to light since that paper was published.

INTRODUCTION
Anti-unpacking tricks come in different forms, depending on what kind of unpacker they are intended to attack. Unpackers can be in the form of memory-dumpers, debuggers, or emulators:

- A memory-dumper dumps the process memory of the running process without regard to the code inside it.
- A debugger attaches to the process, allowing single-stepping, or the placing of breakpoints at key locations, in order to stop execution at the right place. The process can then be dumped with more precision than a memory-dumper alone.
- An emulator, as referred to within this paper, is a purely software-based environment, most commonly used by anti-malware software. It places the file to execute inside the environment and watches the execution for particular events of interest.

There are corresponding tricks for each of the above, and they will be discussed separately.

1. ANTI-DUMPING

1.1 Self-unmapping

The data that fills the image space of a process is simply a mapped view of the file. This view can be unmapped using the kernel32 UnmapViewOfFile() function. If the data is moved first to another location, and then any absolute references are adjusted according to the new image base value, then execution can be resumed from the other location. The result is a process that cannot be dumped by ordinary means.

Example code looks like this:
```assembly
push 0
call GetModuleHandleA
mov ebx, [eax+3ch] ; lfanew
```

2. ANTI-DEBUGGING

2.1 Special APIs

2.1.1 NtYieldExecution

The ntdll NtYieldExecution() function is used to allow the currently running thread to give up the rest of its execution period and allow any other scheduled thread to execute. The function returns an error if no threads are scheduled to execute. When an application is being debugged, the act...
of single-stepping through the code causes debug events, and as a result the debugger thread is always scheduled to resume execution. This fact can also be used to infer the presence of a debugger, though it may also detect the presence of a thread that is running with high priority.

Example code looks like this:

```
push 20h
pop ebp
l1: push 0fh
call Sleep
call NtYieldExecution
cmp al, 1
adc ebx, ebx
dec ebp
jne l1
inc ebx
je being_debugged
```

This technique is used by the Extreme Debugger Detector.

### 2.1.2 NtSetLdtEntries

Perhaps because the local descriptor table (LDT) is not used by Windows, it is generally not supported properly (or at all) by debuggers. As a result, it can be used as a simple anti-debugger technique. Specifically, a new LDT entry can be created, which maps to some code. Then, by performing a far transfer of control (call or jump) to the new LDT entry, the debugger might become lost or refuse to go further.

Example code looks like this:

```
;base must be <= PE->ImageBase
;but no need for 64kb align
base equ 12345678h
;sel must have bit 2 set
;CPU will set bits 0 and 1
;even if we don't do it
sel equ 777h
xor eax, eax
push eax
push eax
push eax
push eax
;4k granular, 32-bit
;present, DPL3, exec-only code
;limit must not touch kernel mem
;calculate carefully to use APIs
push (base and 0ff000000h) \ 
  + 0clf800h \ 
  + ((base shr 10h) and 0ffh)
push (base shl 10h) + 0ffffh
push sel
call NtSetLdtEntries
;jmp far sel:l1
db 0eh
dd offset l1 - base
dw sel
l1: ;execution continues here
;but using LDT selector
...```

Turbo Debug32 fails to disassemble the code inside the LDT range, but execution continues correctly. OllyDbg refuses to continue execution within the LDT range. WinDbg disassembles the code correctly inside the LDT range, and execution continues correctly. This technique is used by some malware. It was probably based on some inaccurate documentation on the ReactOS site [2], which misplaces the System bit and includes too many bits in the Type field.

### 2.1.3 NtQueryInformationProcess

The ntdll NtQueryInformationProcess() function has the following parameters: HANDLE ProcessHandle, PROCESSINFOCLASS ProcessInformationClass, PVOID ProcessInformation, ULONG ProcessInformationLength and PULONG ReturnLength. Windows Vista supports 45 classes of ProcessInformationClass information (up from 38 classes in Windows XP), but so far only four of them have been documented by Microsoft. One of these is the ProcessDebugPort. It is possible to query for the existence (but not the value) of the port. The return value is 0xffffffff if the process is being debugged. Internally, the function queries for the non-zero state of the EPROCESS->DebugPort field. A common method for hiding the debugger process from the ProcessDebugPort class is to zero the value, but without checking the process handle for which the presence of the port is being queried. This presents a problem when a debugger is supposed to be present (for example, in Armadillo), since a debug port should exist for that process. This problem has been disclosed publicly [3].

Example code looks like this:

```
xor ebx, ebx
mov ebp, offset l1
call GetStartupInfoA
;sizeof(PROCESS_INFORMATION)
sub esp, 10h
push esp
push ebp
push ebx
push ebx
push ebx
push ebx
push ebx
push ebx
push 1 ;DEBUG_PROCESS
push ebx
push ebx
push ebx
push ebx
push ebx
push offset l2
call CreateProcessA
pop eax
push eax
mov ecx, esp
push 0
push 4 ;ProcessInformationLength
push ecx
push 7 ;ProcessDebugPort
```

Turbo Debug32 fails to disassemble the code inside the LDT range, but execution continues correctly. OllyDbg refuses to continue execution within the LDT range. WinDbg disassembles the code correctly inside the LDT range, and execution continues correctly. This technique is used by some malware. It was probably based on some inaccurate documentation on the ReactOS site [2], which misplaces the System bit and includes too many bits in the Type field.
push eax
call NtQueryInformationProcess
pop eax
test eax, eax
je being_faked
...
;sizeof(STARTUPINFO)
l1: db 44h dup (?)
l2: db "myfile", 0

2.1.4 CloseHandle
As with an invalid handle, if a protected handle is passed to the kernel32 CloseHandle() function (or directly to the ntdll NtClose() function) and no debugger is present, then an error code is returned. However, if a debugger is present, an EXCEPTION_HANDLE_NOT_CLOSABLE (0xC0000235) exception will be raised. This exception can be intercepted by an exception handler, and is an indication that a debugger is running.

Example code looks like this:
xor eax, eax
push offset being_debugged
push d fs:[eax]
mov fs:[eax], esp
push eax
push eax
push 3 ;OPEN_EXISTING
push eax
push eax
push $00000000h ;GENERIC_READ
push offset l1
call CreateFileA
push eax
;HANDLE_FLAG_PROTECT_FROM_CLOSE
push 2
push -1
xchg ebx, eax
call SetHandleInformation
push ebx
 call CloseHandle
...
l1: db "myfile", 0

Defeating this method is easiest on Windows XP, where a FirstHandler Vectored Exception Handler can be registered by the debugger to hide the exception and silently resume execution. Of course, there is the problem of hooking the kernel32 AddVectoredExceptionHandler() function transparently, in order to prevent another handler from registering as the first handler. However, it is still easier than transparently hooking the ntdll NtClose() function on Windows NT and Windows 2000 in order to register a Structured Exception Handler to hide the exception. This method has been disclosed publicly [4].

2.1.5 NtSystemDebugControl
The ntdll NtSystemDebugControl() function could have been a very nice function for detecting debuggers. It was introduced in Windows NT, and its capabilities increased in subsequent versions of Windows. It supported a SysDbgQueryModuleInformation command, which was an alternative to the SystemProcessInformation class of the ntdll NtQuerySystemInformation() function. Windows XP introduced the SysDbgReadVirtual command, which allowed the reading of virtual memory from anywhere in the system. There were other commands for writing to virtual memory, reading and writing physical memory and MSRs, among others. Alas, in Windows 2003 SP1 and later, all of these functions were disabled. The functions that remain are for enabling and disabling the kernel debugger, and querying and setting some minor behaviours.

2.1.6 Non-continuable exceptions
When an exception occurs, the flags specify whether it is continuable or not. A continuable exception is one for which the cause can be corrected, and then execution can resume from the location at which the exception occurred. An example of a continuable exception is a memory access violation. The use of such an exception is how user-mode paging is implemented by packers such as Shrinker.

A non-continuable exception is one for which, under normal circumstances, the cause cannot be corrected. An example of a non-continuable exception is a division by zero. Any attempt to resume execution after a non-continuable exception is raised will cause Windows to issue an EXCEPTION_INVALID_DISPOSITION exception, prior to terminating the application. However, through the use of a breakpoint in the ntdll RtlRaiseException() function, it is possible to interfere with that sequence. Specifically, the context that is saved on the stack prior to the breakpoint exception can be altered to clear the non-continuable flag. Once that is done, execution of the ntdll RtlRaiseException() function can be resumed. At that point, the call to ntdll RtlRaiseException() becomes indistinguishable from an ordinary call. EXCEPTION_INVALID_DISPOSITION is still delivered to the exception handler, but the application is no longer terminated upon return from the handler. Further, the cause can be corrected. In the case of a division by zero, the divisor can be replaced with a non-zero value, and then the division can be attempted again. This technique has been disclosed publicly [5].

2.2 Hardware tricks
Besides the prefetch queue trick that was described in [1], there is another trick that detects single-stepping. It has worked since the earliest of Intel CPUs, and was common in DOS, but it still works in all versions of Windows. The trick relies on the fact that certain instructions cause all interrupts to be disabled for one instruction. In particular, loading the SS register clears interrupts for one instruction in order to
allow the [E]SP register to be used without risk of stack corruption. Of course, the [E]SP register does not need to be loaded. Any instruction can follow the load of the SS register. If a debugger is being used to single-step through the code, then the T flag will be set in the EFLAGS image. This is typically not visible because a debugger will clear the image whenever the flags are saved. However, if the debugger cannot gain control in time to intercept the save, then it has no way of hiding the T flag. Specifically, the debugger cannot gain control if all interrupts are disabled.

Example code looks like this:

```
push ss
pop ss
pushfd
```

```
test b [esp+1], 1
jne being_debugged
```

This technique is used by the ChupaChu debugger test.

### 2.3 Device names

Tools that make use of kernel-mode drivers also need a way of communicating with those drivers. A very common method is through the use of named devices. Any success when attempting to open such a device indicates the presence of the driver.

Example code looks like this:

```
xor eax, eax
mov edi, offset l2
l1: push eax
    push eax
    push 3 ;OPEN_EXISTING
    push eax
    push eax
    push eax
    push edi
    call CreateFileA
    inc eax
    jne being_debugged
    or ecx, -1
    repne scasb
    cmp [edi], al
    jne l1
...
```

```
l2: <array of ASCIIZ strings, null to end>
```

Recent lists include the following names:

- \SPCommand
- \Syser
- \SyserBoot
- \SyserLanguage
- \SyserDbgMsg

These names belong to Syser.

### 2.4 OllyDbg-specific

A potential arbitrary code execution vulnerability was reported recently for OllyDbg [6]. However, the author of this paper discovered that the report is incorrect with respect to the target. The problem is not in OllyDbg, but in the dbghelp.dll that OllyDbg carries. This DLL is used by multiple debuggers, including IDA, SoftICE and WinDbg, but not Turbo Debug32 or Immunity Debugger, for example. It is also shipped as part of Windows itself. The version of the DLL that OllyDbg carries is indeed vulnerable to a stack buffer overflow. The problem was introduced in Windows XP, and corrected in Windows Server 2003. In Windows 2000, where the file was introduced, a fixed-length copy was performed. Since this was unnecessarily large in most cases, and had the potential to cause crashes because of out-of-bounds reads, it was replaced in Windows XP with a string copy. However, the length of the string was not verified prior to performing the copy, leading to the buffer overflow. The fix was to use a string copy with a specified maximum length.

There is a little-known problem in OllyDbg’s analyser of floating-point instructions, which is caused by having no mask on invalid floating-point operation errors [7]. This allows two special values to cause floating-point errors when converting from double-extended precision to integer. The values are +/- 9.2233720368547758075e18. There is a publicly available demonstration that specifies only the positive value as a candidate, without mentioning that the negative value is also a candidate.

Example code looks like this:

```
fl d t [offset l1]
...
```

```
l1: dq -1
dw 403dh
```

This is the public version. The alternative code looks like this:

```
fl d t [offset l1]
...
```

```
l1: dq -1
dw 0c03dh
```

### 2.5 SoftICE interrupt 0x2D denial of service

When the kernel32 OutputDebugString() function is called and no user-mode debugger is present, eventually the following code is executed:

```
mov eax, [ebp+8] ;service (1)
    ;pointer to message structure
mov ecx, [ebp+0ch]
mov edx, [ebp+10h] ;unused (0)
int 2dh
```

If SoftICE is installed, then the DbgMsg.sys driver is loaded, even if SoftICE isn’t running. DbgMsg.sys hooks...
interrupt 0x2D and executes this code when interrupt 0x2D is executed:

```
    push ecx
    push eax
    call l1
    ...
    ;message structure
l1:  mov ecx, [esp+8]
    mov eax, [eax+4] ;bug here
```

The read from [eax+4] without checking first if the pointer is valid, leads to a kernel-mode crash (blue screen) if the original ECX register value is invalid.

Example code looks like this:

```
    push 1
    pop eax
    xor ecx, ecx
    int 2dh
```

This interrupt 0x2D bug has already been published [8], but without any details about exactly why it happens.

### 3. ANTI-EMULATING

#### 3.1 Hardware tricks

##### 3.1.1 Exception priority

Given the following code:

```
xor eax, eax
push offset l1
push d fs:[eax]
mov fs:[eax], esp
push -1
popfd
;force an exception to occur
mov eax, [eax]
;ExceptionRecord
l1: mov eax, [esp+4]
    ;ExceptionCode
    mov eax, [eax]
```

What is the value of EAX when l2 is reached? Perhaps surprisingly, it is not EXCEPTION_ACCESS_VIOLATION (0xC0000005). It is EXCEPTION_SINGLE_STEP (0x80000004). What actually happens is that an EXCEPTION_ACCESS_VIOLATION is raised, but its delivery to the debuggee is delayed. The reason is that the trap flag remains set when the ntdll KiUserExceptionDispatcher() function gains control, and then after the first instruction is executed, the single step exception is raised and delivered to the debuggee. Upon returning from the debuggee, after dealing with the EXCEPTION_SINGLE_STEP, the EXCEPTION_ACCESS_VIOLATION is delivered to the debuggee. This technique has been disclosed publicly [9].

### 3.2 Software interrupts

#### 3.2.1 Interrupt 0x2E

Interrupt 0x2E is an interface for user-mode code to communicate with native kernel-mode APIs. It was introduced in Windows NT, and continues to be supported for compatibility reasons. The interface accepts a function index in the EAX register. A bounds check is performed against this index, prior to dispatching the request if it is valid. However, if the index is out of bounds, then Windows will return a STATUS_INVALID_SYSTEM_SERVICE (0xC000001C) value in the EAX register. If the index is within bounds, then a bounds check is performed against the parameter pointer in the EDX register. If the parameter pointer is out of bounds, then Windows will return a STATUS_ACCESS_VIOLATION (0xC0000005) value in the EAX register. Some emulators do not support this behaviour.

Example code looks like this:

```
or eax, -1
    cdq
    int 2eh
    int 2eh
    cmp eax, 0c0000005h
    jne being_debugged
```

This technique is used by the ChupaChu debugger test.

### CLOSING REMARKS

Anti-unpacking tricks continue to be developed because the older ones are constantly being defeated. In part two of this series next month, we will describe some tricks that might become common in the future, along with some countermeasures.

The text of this paper was produced without reference to any Microsoft source code or personnel.

### REFERENCES

REPERCUSSIONS OF DYNAMIC TESTING
Roel Schouwenberg
Kaspersky Lab, USA

Anyone who is remotely involved in the anti-malware industry will know that testing is a hot topic – the subject has received a lot of attention lately, particularly following the formation of AMTSO, the Anti-Malware Testing Standards Organization, earlier this year.

This article is not intended to be the nth paper describing how testing should be performed. However, it will highlight one potential consequence of the attention currently being devoted to dynamic testing: the possibility that the increased focus on dynamic testing may inspire malware authors to devote more attention to circumventing products’ protection capabilities, rather than just their detection abilities.

This article will use a number of examples and scenarios to evaluate the risk associated with dynamic testing. It will also put forward a number of suggestions for ways in which testers can mitigate the risk.

TESTING METHODS

Before evaluating the risks associated with dynamic testing we will first have a look at a number of other testing methods.

Static testing

Static testing is the most straightforward type of testing: an on-demand scan is run on a collection of malware. In order to produce meaningful results, any respectable static test these days must use a collection of malware containing thousands of samples, while the test sets used by testing bodies AV-Test and AV-Comparatives usually contain hundreds of thousands of samples and in some cases more than a million samples.

Since the test collections are so large, the results contain little (if any) useful detail that malware authors can use to help them fine-tune their creations.

Another thing to bear in mind when conducting static tests is the way in which the test collection is broken down. Certain less credible tests seem to have been performed on a big pile of unsorted malware samples – and although the tester may have internally enumerated the results of different sub test sets, this would still be considered bad testing practice.

Other, more credible tests make proper differentiation in their published test results. They sort their test collections into sub-sets – such as viruses, worms and trojans – and publish results for each sub-set. Some tests go even further and attempt to differentiate, for instance, between backdoors and spyware trojans.

Although this provides a greater level of detail, it still does not provide much useful information for the malware author. The only possible risk from static tests comes from those that look at how well products detect polymorphic/metamorphic malware. Such tests may highlight weaknesses in certain products.

However, polymorphic/metamorphic malware is inherently more difficult to detect than static malware – and a malware author who needs the results of a test to find this out is most likely not competent enough to write such malware anyway. Having said that, someone who is not competent enough to write a polymorphic piece of malware can now go to the underground and either buy such a piece of code or advertise for someone to create it.

Response time testing

Although response time testing is carried out only rarely these days, it is still worth looking at. It was most popular during the era of the big email worm outbreaks – NetSky, Bagle, Mydoom, Sober and Sobig are classic examples of that period.

In contrast to static testing, the size of the sample sets used for response time testing are very small. One type of response time testing measures the overall performance of AV vendors based on a larger, though still relatively small, test set [1] – this type doesn’t show the results for individual samples. The second type of testing measures detection on a per-sample basis [2].

Such specific, per-sample results could be a clear aid to malware authors. One can speculate that the published response time test results may have led certain malware authors to change their approach to make detection of their malware harder. One example is that of W32/Sober.K [3], which appended random garbage at the end of its file during installation in a deliberate attempt to slow down AV detection. It is quite possible that the author introduced this functionality after being unhappy with the response time results of earlier Sober variants.

Today’s response time testing results are published in a more generic fashion with little reference to specific samples. The risk of malware authors gaining too much information is therefore very low.

Retrospective testing

In retrospective testing, an out-of-date product is tested against up-to-date malware samples – but, other than its purpose (to investigate the product’s ability to detect
unknown samples) and the age of the updates, retrospective testing is no different from static testing.

The level of risk that retrospective tests introduce is very low, just like regular static tests.

**Dynamic testing**

In dynamic testing, malware samples are introduced into the test system with the intent to execute them. Only a small number of samples can be used in these tests because the process of executing each one is very time consuming. AMTSO has published a document which explains the idea behind dynamic testing in greater detail [4].

Ideally, the samples are introduced onto the test system in the ‘right’ way – for instance via drive-by download. Even when automated, this is a very time-consuming task, and the fact that virtual machines need to be avoided in order to obtain valid test results does not help the matter.

The number of samples included in tests is currently in the dozens. In time, with better hardware and optimized processes, we can expect the number of samples being used to reach the hundreds. However, the risk to which the industry is exposed with the introduction of dynamic testing is far greater than that associated with any of the other popular testing methods.

A large part of the industry is spending a lot of time on educating people about (new) proactive technologies, and AMTSO has put a lot of work into compiling a best practices paper for dynamic testing [4]. What is collectively being said is that the focus of testers on products’ detection rates alone is outdated, and testers now need to consider their protective measures as well.

There’s little doubt that the malware authors are also listening.

**TIMES CHANGE**

Some five years ago, Symantec incorporated a protection feature called ‘anti-worm’ into its Norton products. This was a behavioural system used to catch email worms proactively. Symantec expected to have to update the technology no more than six months after its introduction in order for it to remain effective against evolving malware. However, the developers have never had to update it [5]. Malware authors either did not know about the technology, didn’t bother to circumvent it, or were unable to.

In May 2006, Kaspersky Lab launched its version 6 product line which featured a new-generation behaviour blocker. Six months later a patch had to be released for the behaviour blocker because new variants of the LdPinch trojan family were bypassing the technology that had initially been capable of catching them.

What was the difference between the two behaviour blockers? ‘Anti-worm’ was introduced in the era of big malware epidemics, driven mostly by fame-seeking authors. By 2006 the majority of malware out there was being driven by profit, including LdPinch. Another thing to bear in mind is that Kaspersky Lab is a Russian company and LdPinch was a Russian creation, targeted mainly at the Russian market.

However, there could be a third explanation for the difference (though it should be noted that the author of this article is by no means a marketing expert); it would seem that Kaspersky Lab invested more effort into the marketing of its behaviour blocker than Symantec did at the time it introduced ‘anti-worm’, thus malware authors were more aware of the Kaspersky product and made the effort to circumvent it.

**MULTI-SCANNERS**

Multi-scanners are another interesting demonstration of malware authors using legitimate services to gain information for their own purposes. Online multi-scanners enjoy great popularity these days, with JottiScan [6] and VirusTotal [7] being the most popular.

These websites provide a service whereby the user can upload a file and have it scanned by a whole range of scanners to see what the various products detect it as (and if they detect it at all).

These services also enjoy some popularity with malware authors who use them to test their latest creations and see whether they successfully avoid detection. While some anti-malware vendors provide the multi-scanner sites with their most recent scanners and ask the maintainer to use the most paranoid settings to detect as much malware as possible, there are also vendors who don’t offer their most recent scanner to these sites. They may also ask the maintainer to use lower than maximum settings, so the product will detect less malware than it is capable of in real-world use [8], thus not revealing its true detection capabilities.

**REPERCUSSIONS FROM THE UNDERGROUND**

These days malware that bypasses protection features is by no means rare. However, the vast majority of malware authors still focus solely on bypassing detection mechanisms.

Some anti-malware products have little in the way of protection measures, and bypassing their detection
capabilities still means bypassing the entire product. Therefore it’s not so strange to see Win32 PE malware samples that are obfuscated in such a way that de-obfuscation takes roughly two minutes on a Core 2 Duo running at 2,500 MHz. However, the same malware samples can be detected proactively using behaviour blocker technology from two years ago [9].

There’s little doubt that the current noise regarding protection technologies and dynamic testing is causing some malware authors to pay extra attention to these types of technology. With the current buzz surrounding the topic, it’s likely that the interest of more malware authors will be piqued.

There are a number of scenarios likely to occur: firstly, it’s likely that new groups will form in the underground that will focus on providing the means for malware to circumvent protection technologies. Secondly, there may be a new market for improved multi-scanners which will test products’ protection technologies as well as their detection abilities.

The big problem with malware bypassing protection technologies is the matter of fixing the holes the malware authors are exploiting. While vendors can ship a new signature update in hours or even minutes, fixing holes takes much longer – we’re talking about a response time of week(s) rather than days, let alone hours.

WHAT CAN TESTERS DO?
Revealing per-sample test result details is a much more dangerous idea with dynamic testing than it is with response time testing. While there is a low-to-moderate risk in revealing too much detail with response time testing, the risk is very high with dynamic testing.

Having a limited sample set for testing means that the samples tested need to be very relevant. If testers are going to publicize the results for each such important sample, including how individual products perform against them, then this is extremely valuable information for the malware authors. It will show them against which products they need to improve their creations.

_Virus Bulletin_ is not yet publishing dynamic testing results, but plans to use information from its (upcoming) prevalence table to pick samples for testing. While the testers will start out just by mentioning malware families, they may end up disclosing specific malware names as well [10].

_AV-Comparatives_ is also not yet publishing dynamic testing results, but intends to publish the names of the samples being used for its future dynamic tests.

As pointed out, this approach should be avoided. _AV-Test_, which is already performing dynamic tests, takes a better approach. Magazines are prohibited from disclosing the malware names or hashes of files that were used in the test. However, _AV-Test_ will share the hashes or samples with the AV vendors that participated in the test [10].

Though slightly less transparent for end-users, this approach is by far preferable in terms of risk mitigation, while also allowing for any vendor to notify the tester if they find that any non-relevant samples have been used in the test set.

CONCLUSIONS
The adoption of dynamic testing brings with it some new challenges. Now, more than ever, testing can have real consequences for malware authors and their actions.

Security vendors need to take care that in their quest for education they do not lose sight of what really is important: the protection of users.

Care must be taken to avoid a situation in which education speeds up malware evolution and causes more problems than solutions. AMTSO’s first rule of the fundamental principles of testing document states that testing must not endanger the public [4].

The attention on protection technologies and dynamic testing is inevitably leading to increased awareness on all parts, including that of malware authors. It will be up to the industry as a whole, possibly in the form of AMTSO, to minimize the risk and ensure that testers do not reveal too much detail in their public test results.

REFERENCES


[9] Exact name of the sample can be obtained by contacting the author of this article.

Frame4 Security Services was set up by my business partner and me in 2006, operating from Alphen aan den Rijn, not far from Amsterdam, in The Netherlands. I handle the business side of things, while my partner is primarily involved in the technical aspects of the business, including the maintenance of the malware database.

There is a slightly Hitchcockian story behind the setting up of the business, as it all started with a discussion on a train. On the train I had met a technical rep from a company that was in the process of adding the finishing touches to a content-filtering device. The rep explained that they had experienced great problems testing the device – because they simply could not get their hands on enough malware. The developers had approached some AV companies, but had been stonewalled. After listening to the man’s experiences I started thinking – what if we could develop a business model that would potentially give security researchers the room to concentrate on developing solutions, instead of spending valuable time trying to track down malware samples?

The result was the MD:Pro malware repository service. Criminals have had access to malware repositories for years, whereas the mainstream security industry has never had a reputable research and development resource – and that is the niche we aimed to fill.

While the anti-virus industry has had file exchange mechanisms in place for many years, the exclusivity of this approach has meant that, for those security providers that fall outside of the core circle of anti-virus vendors, enormous amounts of time, money and effort have to be invested in order to gather resources for R&D and testing purposes.

This is perfectly understandable, of course, as the AV industry invests huge amounts of time and resources in collecting malware – though I am puzzled and frustrated by the tendency for many members of the anti-virus community to look on our company and its services with suspicion and to doubt our ethics. For me it is simple: I believe that the security industry has long needed a service such as MD:Pro, and the amount of interest and positive feedback we have received from the mainstream security community affirms this belief.

Frame4 currently delivers weekly samples via FTP and monthly samples on DVD, though MD:Pro is currently being re-structured in order to build a more efficient system. The new service will concentrate on collecting more samples and providing more information on those samples (for example: multiple AV scanning, advanced file analysis, malware classification, packer identification etc.), along with a secure FTP server from which the samples can be downloaded.

We are aiming to go fully live with the restructured service by the end of 2008/beginning of 2009. The new service will be complemented by a members-only website providing a range of searchable file statistics, RSS feeds, etc., though it will no longer be possible to download live
samples from the website. We have also decided to get rid of the various different subscription options we started with and instead provide one type of membership at a fixed price.

ETHICS

While we do not have an 'ethics statement' per se, we do have a strict set of rules and guidelines within which we operate, and we stick to them rigidly – for example, we only provide malware to corporate customers, and only to those in the IT security field. We made this decision on day one, and we have stuck by it without exception. Since we began operating we have had many requests from individuals wanting to access the database – particularly in the early days – but we have always stuck to our guns and declined to do business with non-corporate customers.

We are lucky in that our customers are generally well-known anti-malware providers or respected players in the IT security field – and if we are dealing with specific individuals within these companies, their identity can easily be verified by contacting the company directly. However, if we are in any way in doubt about a company, its motives (what it is going to do with the malware) or the individual(s) we are dealing with within a company, we will not accept them as a customer – it is simply not worth running the risk of malware getting into the wrong hands. And, yes, we have had to turn away applicants (both individual and corporate) on a few occasions.

TECHNICAL DETAILS

All matters pertaining to the technical side of the business are dealt with by my business partner. When I asked him to describe how our malware collections are maintained, he literally threw the book at me – according to him, Analysis and Maintenance of a Clean Virus Library by Vesselin Bontchev is a must-read.

Our collection currently exceeds 1.7 million malware samples, with between 20,000 and 100,000 new samples being added every month. Our samples come from various sources: our own honeypots, trading with other security providers, strategic alliances with security companies, donations and our own research. We are now even trading samples with an anti-virus company.

All samples that we receive are checked against our database; existing samples are discarded, and new samples are run through a set of tools (AV packages, PEiD, TrID, etc.) that collect various pieces of information about them. This information is written to our database and the identified samples are moved to the repository. Any unidentified and/or suspicious samples are moved to a holding area for further analysis.

Access to the collections for customers that have been approved, is granted on a monthly subscription basis. Customers sign an agreement, pay a subscription fee and start receiving samples from us on a weekly or monthly basis.

The collection is currently subdivided into broad categories by type of malware: worms, viruses, trojans, backdoors and other files (jokes, hoaxes, adware), though we are working on a complete reclassification of the database, as part of the current system overhaul.

CUSTOMERS

Our customer base consists almost entirely of anti-malware vendors – generally those that exist on the fringes of the traditional AV community and in the mainstream security industry. Our malware database is in use in various commercial anti-malware products around the world, ranging from various white and/or blacklisting applications to hardware-based security appliances. DriveSentry, for example, uses the samples and the report we provide along with the malware as the backbone of its innovative blacklisting products. There are also customers who use our samples purely for research purposes – such as the team from Zynamics (run by Halvar Flake), who use the samples to develop and fine-tune their VxClass software.

THE FUTURE

Our hopes for the company and its services in the future are to gain the trust of the entire IT security community – to be acknowledged as a legitimate business and as individuals who are seriously dedicated to the cause and what we believe in. A different approach is not necessarily evil, but in some cases it is a necessary evil. And, of course, we would like MD:Pro eventually to become the world’s biggest and most trusted malware repository.

We want to be able to provide information about malware to all IT security companies who need and want to have access to it. The knowledge about a specific piece of malware should not be exclusive to one company, or a group of companies – if anything, this is counterproductive. There are a lot of brilliant ideas out there about how malware should be tackled, some in development, some yet to be developed and some on the shelf; we believe that there is no reason why research should suffer due to competition or for the pursuit of exclusivity.
COMPARATIVE REVIEW

WINDOWS VISTA X64

John Hawes

The final VB100 of the year sees a double whammy of potential pitfalls for our comparative participants – the Vista operating system, which still seems shiny and new as well as a little scary (to both developers and users), as well as the x64 architecture, whose ostensible compatibility with standard 32-bit software belies oddities and intricacies that developers ignore at their peril. The announcement of the test brought a few surprises, as several regulars opted to skip this one, but the majority of veteran competitors took part as usual, along with several newer faces, many of whom look set to join the ranks of our regulars.

A total of 25 products were expected to take part, however, due to technical difficulties one of our most reliable participants was unable to provide a product on the deadline date. While some vendors have produced dedicated x64 products, many continue to rely on their standard versions. This was expected to cause some difficulties, and after a setup period considerably shaken by a series of hardware disasters, the legacy of a temperature control failure a few weeks ago which continued to cause problems throughout the testing period, we could only hope to get through this month’s test with a modicum of sanity intact.

PLATFORM AND TEST SETS

Once again we visit the 64-bit edition of Microsoft’s Windows Vista, which last played host to a VB100 comparative in August 2007. The user base of the Vista platform continues to grow slowly, with XP still the platform of choice for the vast majority of desktop users – most estimates suggest XP resides on between 70% and 80% of systems, while Vista still runs on less than 20% almost two years after its introduction. This pattern looks likely to change as sales of XP are gradually retired, but hard-core Vista-haters continue to insist they’ll wait it out and see what the next iteration looks like before abandoning XP.

Meanwhile, the x64 architecture, having had a rather longer time to mature, seems to have become the standard for current processors, with straight x86 fading away into the past. The close compatibility between the two, which has helped this growth considerably, means that many continue to use x86 operating systems and not take full advantage of the architecture, while those running full-blown x64 setups expect to find all their 32-bit applications running without difficulty – although, in this area mileage may vary considerably. The AMD64-based hardware used for much of the testing in the VB test lab generally idles along happily running 32-bit operating systems, but once in a while we allow it full rein with a platform designed specifically for the architecture. This is always a cause for concern in VB100 testing, where history has taught us that ‘fully compatible’ doesn’t necessarily mean identical behaviours.

The installation and setup of Vista is fairly straightforward, but was hampered as usual by the Machiavellian activation process and complications porting images from one system to another for maximum similarity. The standard set of tweaks were made to the default installation after applying the recent service pack – drivers for display and networking hardware were added, network shares connected, and users and passwords set up. For the purposes of this test, an admin-level user was used throughout, with the User Access Controls running in their default state – while we anticipated some annoyances from the likely large numbers of pop-ups, it seemed appropriate to monitor how various products integrated with this safety measure.

The test sets meanwhile underwent their usual minor upgrades, with much of our efforts concentrated on broader upgrades across the lab in preparation for more significant changes in upcoming tests. A sizeable chunk of new software was added to the clean set, and the trojan set used in the previous test was retired and replaced entirely with samples gathered in the last three months. We hope to introduce the same pattern of replenishment with fresh samples for the other test sets in time for the next review, along with some entirely new sets, more on which later.

The WildList set was aligned with the September issue of the WildList, which was released towards the end of October, a few weeks prior to the product deadline. The changes since the previous set included the addition of another flood of online gaming password-stealers, and the retirement of large swathes of older material. These included most of the bot families that once dominated, along with significant numbers of worms such as W32/Stration (aka Warezov) and W32/Rontokbro (aka Brontok). Several more variants of W32/Virut also fell from the list, indicating a gradual decline in numbers of a family which has caused more than its fair share of difficulties in detection, but we hope to add some of these variants to the polymorphic set, in greatly increased numbers, to ensure detection for this tricky kind of malware remains up to scratch. A few other, less sophisticated items were added to this set this month.

With minimal changes to our own sets, and expansion of the WildList set sizeable but fairly uniform, there looked to be few potholes for products to trip on this month.
Agnitum Outpost Security Suite Pro 2009
6.5.2358.316.0607

ItW 100.00%  Polymorphic  80.15%
ItW (o/a) 100.00%  Trojans  53.04%
Worms & bots  99.94%  False positives  0

Agnitum has become a regular participant in our tests over the past year or so, and the product has made itself welcome with good design and solid functionality. The installation process sparked a yellow alert from the UAC system, defaulting to cancel, followed by some more warnings which were eliminated by allowing the system to ‘always trust’ Agnitum. With these hurdles bypassed, the installation process took a few minutes followed by a reboot, and we were good to go. The interface is simple and clear, with ample controls and fine-tuning available, and everything seemed to run smoothly with no jerks or lags. Speeds weren’t the best, but false positives were absent across the clean and speed sets. Detection in the WildList set was above reproach, and fairly good elsewhere, although a little less than might be hoped for in the trojans set.

The product features a variety of behavioural protection mechanisms as part of its main component (the highly regarded firewall), so many of the samples missed in the other sets may in fact be protected against in other ways in a real-world setting. Achieving the VB100 requirements without difficulty, Agnitum takes the first award of the month.

AhnLab V3 Internet Security 7.0 Platinum Edition 7.6.4.1

ItW 100.00%  Polymorphic  99.78%
ItW (o/a) 100.00%  Trojans  66.73%
Worms & bots  98.87%  False positives  0

AhnLab’s product only produced a basic alert from the UAC system, and installed rapidly without the need for a reboot. The interface seemed fairly clear and lucid, but this proved to be deceptive, as numerous vital controls are tucked away where you would least expect them. There were some ominous lags when opening logs (perhaps understandably as large amounts of information were involved) but also when accessing the file system browser as part of the manual scan process. When faced with a 25s pause for a simple browse dialog on a fast modern machine, one could be forgiven for suspecting something is wrong.

Scanning speeds reflected this slightly lethargic attitude, but were far from dismal. Detection rates were much better than expected, after the developers have put some hard work into catching up with the polymorphic set in recent months. Across the clean sets, a large number of files were flagged on demand, which seemed particularly odd as many of them were in areas reserved for files accompanying standard Windows installations. Closer inspection of logs showed that the ‘malware’ in question was labelled ‘W97M/Macro’, together with the information that a macro removal tool could be used to remove the offending items. After much consideration and close analysis of the wording of logs, it was decided that, though it was a very close call and some users could be alarmed by it, this intentional detection did not count as a full false alarm. AhnLab thus qualifies for a VB100 award.

Alwil avast! 4.8 Pro

ItW 100.00%  Polymorphic  91.38%
ItW (o/a) 100.00%  Trojans  93.21%
Worms & bots  99.82%  False positives  1

Alwil’s avast! continues to delight and baffle in equal measure, with a lightning-fast install hindered only by the UAC at the start, followed by a reboot and further UAC pop-ups requesting permission to access the interface. This itself remains unchanged, a combination of stylized simple controls with an ‘enhanced’ version for power users. The full control system is a rather ungainly thing which, with the benefit of considerable experience, was eventually wrangled into the required shape and lumbered its way through the tests. On-demand scanning speeds were quite impressive, but on-access speeds somewhat less so.

Detection, on the other hand, was superb, with an excellent score in the new trojans set and even better elsewhere. The WildList presented no difficulties, and in the clean sets a number of files in deep archives were warned against as potential decompression bombs. While these caused no problems, another file was mislabelled as malware which, unfortunately for Alwil, was enough to spoil its chances of a VB100 this month.

AVG Internet Security 8.0.199

ItW 100.00%  Polymorphic  91.74%
ItW (o/a) 100.00%  Trojans  96.10%
Worms & bots  99.96%  False positives  0

AVG is a big player in the free-AV market, soon to be joined by an offering from Microsoft, but the company’s full suite offers an impressive selection of extras. These are made
the most of by a main interface crammed to bursting with buttons advertising the various protective layers available, rendering it somewhat cluttered and overwhelming. The installation is quite a slog, with an initial UAC prompt followed by numerous stages including the offer of Yahoo! Search toolbars, the setting up of various scheduled checks, selection of networking options and so on, before a reboot finally finishes things off.

Once up and running, the busy interface fairly sensibly requires UAC confirmation to get to the on-access controls, and is reasonably well laid out with accessible but less than comprehensive configuration controls. Speeds were very good on access but a little less splendid on demand where things were a little more thorough. False positives were absent, and detection rates again quite excellent across all sets. With no problems in the WildList, AVG wins a VB100 award this month.

**Avira AntiVir Pro 8.2.0.609**

<table>
<thead>
<tr>
<th>WildList viruses</th>
<th>Missed</th>
<th>%</th>
<th>Missed</th>
<th>%</th>
<th>Missed</th>
<th>%</th>
<th>Missed</th>
<th>%</th>
<th>FP</th>
<th>Susp.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agnitum Outpost</td>
<td>0</td>
<td>100.00%</td>
<td>2</td>
<td>99.94%</td>
<td>366</td>
<td>80.15%</td>
<td>1213</td>
<td>52.69%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AhnLab V3 Internet Security</td>
<td>0</td>
<td>100.00%</td>
<td>3</td>
<td>99.87%</td>
<td>51</td>
<td>99.78%</td>
<td>853</td>
<td>66.73%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alwil avast!</td>
<td>0</td>
<td>100.00%</td>
<td>3</td>
<td>99.82%</td>
<td>312</td>
<td>91.38%</td>
<td>172</td>
<td>93.21%</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>AVG Internet Security</td>
<td>0</td>
<td>100.00%</td>
<td>1</td>
<td>99.96%</td>
<td>52</td>
<td>91.74%</td>
<td>155</td>
<td>93.95%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Avira AntiVir</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>55</td>
<td>97.85%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CA eTrust</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>177</td>
<td>92.51%</td>
<td>1415</td>
<td>44.81%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ESET NOD32</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>327</td>
<td>87.25%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fortinet FortiClient</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>1802</td>
<td>29.72%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRISK F-Prot</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>121</td>
<td>96.46%</td>
<td>820</td>
<td>68.02%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>F-Secure Client Security</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>60</td>
<td>98.24%</td>
<td>287</td>
<td>88.81%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kaspersky Anti-Virus</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>60</td>
<td>98.24%</td>
<td>333</td>
<td>87.01%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kingsoft Internet Security</td>
<td>0</td>
<td>100.00%</td>
<td>16</td>
<td>99.27%</td>
<td>1686</td>
<td>34.24%</td>
<td>2068</td>
<td>47.39%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>McAfee VirusScan</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>475</td>
<td>81.47%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Microsoft Forefront</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>130</td>
<td>96.04%</td>
<td>566</td>
<td>77.93%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Microsoft OneCare</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>130</td>
<td>96.04%</td>
<td>553</td>
<td>78.43%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Norman Virus Control</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>1079</td>
<td>71.06%</td>
<td>970</td>
<td>62.17%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Quick Heal AntiVirus</td>
<td>0</td>
<td>100.00%</td>
<td>51</td>
<td>95.53%</td>
<td>986</td>
<td>81.31%</td>
<td>1483</td>
<td>42.16%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rising Antivirus</td>
<td>0</td>
<td>100.00%</td>
<td>5</td>
<td>99.62%</td>
<td>1402</td>
<td>57.22%</td>
<td>1632</td>
<td>36.35%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sophos Anti-Virus</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>158</td>
<td>93.34%</td>
<td>772</td>
<td>69.89%</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>Symantec Endpoint Protection</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>765</td>
<td>70.16%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VirusBuster Professional</td>
<td>0</td>
<td>100.00%</td>
<td>2</td>
<td>99.94%</td>
<td>390</td>
<td>78.21%</td>
<td>1215</td>
<td>52.61%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Webroot I.S. Essentials</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>813</td>
<td>89.83%</td>
<td>806</td>
<td>68.56%</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

Avira is another of the leading players in the free market, if it can be called such, and has an excellent reputation for detection. The product’s installation process was a little less slick, with another of the yellow UAC pop-ups warning of ‘unidentified’ software, and a readme appearing over the
top of a dialog box towards the end. The installation was followed by an attempt to scan the system, which was eventually stopped after some initial difficulties, and testing got under way.

The interface is another of those that appears straightforward but has deceptive moments of illogic, and this was not only apparent on the surface. Several attempts to run scans were found to be failing to access files, an oddity eventually diagnosed as being caused by the on-access scanner preventing the on-demand scanner from working properly. With the appropriate parts disabled, all tests were run through at their usual superb speed and with incredible accuracy. With barely anything missed and not a shadow of a false alarm, Avira justly earns a VB100 award for its product’s performance.

**CA eTrust 8.1.637.0**

<table>
<thead>
<tr>
<th>Malware Type</th>
<th>Missed</th>
<th>%</th>
<th>Missed</th>
<th>%</th>
<th>Missed</th>
<th>%</th>
<th>Missed</th>
<th>%</th>
<th>FP</th>
<th>Susp.</th>
</tr>
</thead>
<tbody>
<tr>
<td>WildList viruses</td>
<td>0</td>
<td>100.00%</td>
<td>2</td>
<td>99.94%</td>
<td>366</td>
<td>80.15%</td>
<td>1204</td>
<td>53.04%</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Worms &amp; bots</td>
<td>0</td>
<td>100.00%</td>
<td>3</td>
<td>99.87%</td>
<td>51</td>
<td>99.78%</td>
<td>853</td>
<td>66.73%</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td>Polymorphic viruses</td>
<td>0</td>
<td>100.00%</td>
<td>1</td>
<td>99.96%</td>
<td>52</td>
<td>91.74%</td>
<td>100</td>
<td>96.10%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trojans</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>33</td>
<td>98.71%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clean sets</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>177</td>
<td>92.51%</td>
<td>1273</td>
<td>50.35%</td>
</tr>
<tr>
<td>CA eTrust</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>2</td>
<td>99.51%</td>
<td>1273</td>
<td>50.35%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ESET NOD32</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>227</td>
<td>91.15%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRISK F-Prot</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>1802</td>
<td>29.72%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>F-Secure Client Security</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>121</td>
<td>96.46%</td>
<td>769</td>
<td>70.01%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kaspersky Anti-Virus</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>60</td>
<td>98.24%</td>
<td>279</td>
<td>89.12%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AVG Internet Security</td>
<td>0</td>
<td>100.00%</td>
<td>16</td>
<td>99.27%</td>
<td>1668</td>
<td>34.95%</td>
<td>2068</td>
<td>47.39%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CA eTrust</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>177</td>
<td>92.51%</td>
<td>1273</td>
<td>50.35%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Microsoft Forefront</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>0.00%</td>
<td>448</td>
<td>82.53%</td>
</tr>
<tr>
<td>Microsoft OneCare</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>130</td>
<td>96.04%</td>
<td>381</td>
<td>85.01%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Norman Virus Control</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>130</td>
<td>96.04%</td>
<td>483</td>
<td>81.16%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Quick Heal AntiVirus</td>
<td>0</td>
<td>100.00%</td>
<td>48</td>
<td>95.79%</td>
<td>986</td>
<td>81.31%</td>
<td>1114</td>
<td>56.55%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rising Antivirus</td>
<td>0</td>
<td>100.00%</td>
<td>3</td>
<td>99.80%</td>
<td>1332</td>
<td>60.80%</td>
<td>1180</td>
<td>53.98%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sophos Anti-Virus</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>158</td>
<td>93.34%</td>
<td>734</td>
<td>71.37%</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>Symantec Endpoint Protection</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>753</td>
<td>70.63%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VirusBuster Professional</td>
<td>0</td>
<td>100.00%</td>
<td>2</td>
<td>99.94%</td>
<td>390</td>
<td>78.21%</td>
<td>1201</td>
<td>53.16%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Webroot I.S. Essentials</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>100.00%</td>
<td>813</td>
<td>89.83%</td>
<td>799</td>
<td>68.84%</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

CA’s product has remained unchanged over several VB100 tests, with the same main installer used each time and simple updates provided for each test. The lengthy installation process with its multiple EULAs runs through on automatic, after an initial
UAC prompt, and was only enlivened this time by a failing updater – an error was diagnosed thanks to the ‘x86’ in the file’s name, rather than from the rather misleading error message, and with a 64-bit version duly replacing it things moved along.

The interface is something of a horror – this time it was less sluggish to respond than usual, but still awkward and fiddly, with access to logging data almost impossible. Some of the functions continue to bemuse, such as the engine selection button which continues to hang around years after the product’s optional second engine was dropped, and the conspicuous lack of archive scanning on access despite clear options to enable it. Nevertheless, scanning speeds remain lightning-fast, and detection rates decent, although a little poor on the trojans set. With no false positives and no items missed in the WildList set, CA earns yet another VB100 award.

**ESET NOD32 Antivirus 3.0.672.0**

<table>
<thead>
<tr>
<th></th>
<th>ItW</th>
<th>Polymorphic</th>
<th>ItW (o/a)</th>
<th>Trojans</th>
<th>Worms &amp; bots</th>
<th>False positives</th>
</tr>
</thead>
<tbody>
<tr>
<td>ItW</td>
<td>100.00%</td>
<td></td>
<td>100.00%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ItW (o/a)</td>
<td>100.00%</td>
<td></td>
<td></td>
<td>91.15%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Worms &amp; bots</td>
<td>100.00%</td>
<td>False positives</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The latest iteration of *ESET’s* product started with another fairly straightforward install, despite dragging on somewhat during the file-copying phase and with a UAC prompt halfway through. It retains its stylish good looks and decent navigability, along with speeds which seem slightly less impressive than in previous versions but still well ahead of the crowd. Some sensible defaults and comprehensive options made for easy testing. The performance was marred by a few buggy moments, the occasional refusal to cooperate and on a couple of occasions full-on freezes, requiring a reboot to regain access to the controls. There were also a few occasions where options appeared to respond in ways not entirely expected.

All this did little to dent an otherwise solid performance, and detection rates were solid with high marks across the board. No trouble with the WildList samples and no false positives means that yet another VB100 award is earned by *ESET*.

**Fortinet FortiClient 3.0.606**

<table>
<thead>
<tr>
<th></th>
<th>ItW</th>
<th>Polymorphic</th>
<th>ItW (o/a)</th>
<th>Trojans</th>
<th>Worms &amp; bots</th>
<th>False positives</th>
</tr>
</thead>
<tbody>
<tr>
<td>ItW</td>
<td>100.00%</td>
<td></td>
<td>100.00%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ItW (o/a)</td>
<td>100.00%</td>
<td></td>
<td></td>
<td>29.72%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Worms &amp; bots</td>
<td>100.00%</td>
<td>False positives</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*FortiClient* brought up another of the yellow UAC prompts during its installation, and some scarier red ones as various driver components were installed. Once the reboot was completed some more confirmation requests were presented when going through the setup process and accessing configuration. Some extremely thorough defaults meant little of this was required, but slowed down the speed tests somewhat. Nevertheless, respectable scanning speeds were evident.

Detection rates were mostly excellent, although in the trojans set the rate dropped sharply; liaison with the developers after a similar performance in the last test suggested many of the items included in the set are covered by the product’s greyware detections, not enabled by default. However, a rescan with these settings turned on produced few extra detections.

Despite this, the WildList was covered just fine, and without false positives *Fortinet* also wins a VB100 award.

**F-Secure Client Security 8.00 build 232**

<table>
<thead>
<tr>
<th></th>
<th>ItW</th>
<th>Polymorphic</th>
<th>ItW (o/a)</th>
<th>Trojans</th>
<th>Worms &amp; bots</th>
<th>False positives</th>
</tr>
</thead>
<tbody>
<tr>
<td>ItW</td>
<td>100.00%</td>
<td></td>
<td>100.00%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ItW (o/a)</td>
<td>100.00%</td>
<td></td>
<td></td>
<td>89.12%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Worms &amp; bots</td>
<td>100.00%</td>
<td>False positives</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*F-Secure* returns us to the more complex world of multi-layer protection, the product including the company’s new and much-vaunted *Deepguard* system, using an online reputation database in addition to local information as part of the behavioural protection system. Sadly, the impact of this could not be fully analysed in our current test setup, but...
the rest of the product seemed pretty solid for the most part.

The installer runs through nice and simply, with a UAC prompt at the start and the selection of local or remote management the only non-standard moments. Once installed, and after a reboot, testing proceeded fairly slowly, thanks to the in-depth multi-engine approach, and the only blot on the performance was a loss of connectivity between various parts of the product at one stage – attempts to set off an on-demand scan were met with messages telling us the ‘AV handler’ was not running. Another reboot soon fixed this, and the problem did not recur. We powered through

<table>
<thead>
<tr>
<th>Product</th>
<th>ACE</th>
<th>CAB</th>
<th>JAR</th>
<th>LZH</th>
<th>RAR</th>
<th>TGZ</th>
<th>ZIP</th>
<th>ZIP-SFX</th>
<th>EXT*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agnitum Outpost</td>
<td>OD 2</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>OA</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>AhnLab V3 Internet Security</td>
<td>OD</td>
<td>X</td>
<td>9</td>
<td>X</td>
<td>9</td>
<td>X</td>
<td>9</td>
<td>X</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>OA</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>√</td>
</tr>
<tr>
<td>Alwil avast!</td>
<td>OD</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
</tr>
<tr>
<td></td>
<td>OA</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
</tr>
<tr>
<td>AVG Internet Security</td>
<td>OD</td>
<td>X</td>
<td>√</td>
<td>X</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>X/N</td>
</tr>
<tr>
<td></td>
<td>OA</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>√</td>
</tr>
<tr>
<td>Avira AntiVir</td>
<td>OD</td>
<td>√</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
</tr>
<tr>
<td></td>
<td>OA</td>
<td>X</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
</tr>
<tr>
<td>CA eTrust</td>
<td>OD</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>OA</td>
<td>X</td>
<td>X</td>
<td>1</td>
<td>X</td>
<td>X</td>
<td>1</td>
<td>X</td>
<td>√</td>
</tr>
<tr>
<td>ESET NOD32</td>
<td>OD</td>
<td>X</td>
<td>√</td>
<td>X</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>OA</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>√</td>
</tr>
<tr>
<td>Fortinet FortiClient</td>
<td>OD</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>OA</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>4</td>
</tr>
<tr>
<td>FRISK F-Prot</td>
<td>OD</td>
<td>1</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>OA</td>
<td>1</td>
<td>X</td>
<td>2</td>
<td>X</td>
<td>X</td>
<td>2</td>
<td>X</td>
<td>√</td>
</tr>
<tr>
<td>F-Secure Client Security</td>
<td>OD</td>
<td>X/N</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>OA</td>
<td>X/N</td>
<td>X/5</td>
<td>X/5</td>
<td>X/5</td>
<td>X/5</td>
<td>X/2</td>
<td>X/5</td>
<td>X/5</td>
</tr>
<tr>
<td>Kaspersky Anti-Virus</td>
<td>OD</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>OA</td>
<td>X/4</td>
<td>X/4</td>
<td>X/4</td>
<td>X/4</td>
<td>X/5</td>
<td>X/1</td>
<td>X/2</td>
<td>X/1</td>
</tr>
<tr>
<td>Kingsoft Internet Security</td>
<td>OD</td>
<td>X</td>
<td>X</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>OA</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>√</td>
</tr>
<tr>
<td>McAfee VirusScan</td>
<td>OD</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>OA</td>
<td>X/2</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
<td>X/N</td>
</tr>
<tr>
<td>Microsoft Forefront</td>
<td>OD</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>OA</td>
<td>X</td>
<td>X</td>
<td>1</td>
<td>X</td>
<td>X</td>
<td>1</td>
<td>1</td>
<td>√</td>
</tr>
<tr>
<td>Microsoft OneCare</td>
<td>OD</td>
<td>X</td>
<td>X</td>
<td>1</td>
<td>X</td>
<td>X</td>
<td>1</td>
<td>1</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>OA</td>
<td>X</td>
<td>X</td>
<td>1</td>
<td>X</td>
<td>X</td>
<td>1</td>
<td>1</td>
<td>√</td>
</tr>
<tr>
<td>Norman Virus Control</td>
<td>OD</td>
<td>X</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>OA</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>√</td>
</tr>
<tr>
<td>Quick Heal AntiVirus</td>
<td>OD</td>
<td>X/2</td>
<td>X/5</td>
<td>2/5</td>
<td>X</td>
<td>2/5</td>
<td>X/1</td>
<td>2/5</td>
<td>X/5</td>
</tr>
<tr>
<td></td>
<td>OA</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Rising Antivirus</td>
<td>OD</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>OA</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>Sophos Anti-Virus</td>
<td>OD</td>
<td>X</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>OA</td>
<td>X</td>
<td>X/5</td>
<td>X/5</td>
<td>X/5</td>
<td>X/5</td>
<td>X/5</td>
<td>X/5</td>
<td>X/5</td>
</tr>
<tr>
<td></td>
<td>OA</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X/N</td>
<td>X/N</td>
</tr>
<tr>
<td>Webroot I.S. Essentials</td>
<td>OD</td>
<td>X</td>
<td>√</td>
<td>5</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>5</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>OA</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>√</td>
</tr>
</tbody>
</table>

Key:
- X - Archive not scanned
- √ - Archives scanned to depth of 10 or more levels
- [1-9] - Archives scanned to limited depth
- X/N - Default settings/thorough settings
- *Executable file with randomly chosen extension
the infected sets with excellent detection rates, scored no false positives in the clean sets and covered the WildList flawlessly. Another VB100 goes to F-Secure for the product’s performance.

**Kaspersky Anti-Virus 2009 8.0.0.454**

<table>
<thead>
<tr>
<th>On-demand throughput (MB/s)</th>
<th>Archive files</th>
<th>Binaries and system files</th>
<th>Media and documents</th>
<th>Other file types</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Default settings</td>
<td>All files</td>
<td>Default settings</td>
<td>All files</td>
</tr>
<tr>
<td></td>
<td>Time (s)</td>
<td>Throughput (MB/s)</td>
<td>Time (s)</td>
<td>Throughput (MB/s)</td>
</tr>
<tr>
<td>Agnitum Outpost</td>
<td>942</td>
<td>3.24</td>
<td>942</td>
<td>3.24</td>
</tr>
<tr>
<td>AhnLab V3 Internet Security</td>
<td>1683</td>
<td>1.82</td>
<td>1683</td>
<td>1.82</td>
</tr>
<tr>
<td>Alwil avast!</td>
<td>34</td>
<td>89.86</td>
<td>612</td>
<td>4.99</td>
</tr>
<tr>
<td>AVG Internet Security</td>
<td>1253</td>
<td>2.44</td>
<td>1253</td>
<td>2.44</td>
</tr>
<tr>
<td>Avira AntVir</td>
<td>265</td>
<td>11.53</td>
<td>289</td>
<td>10.57</td>
</tr>
<tr>
<td>ESET NOD32</td>
<td>756</td>
<td>4.04</td>
<td>756</td>
<td>4.04</td>
</tr>
<tr>
<td>Fortinet FortiClient</td>
<td>286</td>
<td>10.68</td>
<td>286</td>
<td>10.68</td>
</tr>
<tr>
<td>FRISK F-Pro</td>
<td>293</td>
<td>10.43</td>
<td>293</td>
<td>10.43</td>
</tr>
<tr>
<td>F-Secure Client Security</td>
<td>1397</td>
<td>2.19</td>
<td>1852</td>
<td>1.65</td>
</tr>
<tr>
<td>Kaspersky Anti-Virus</td>
<td>591</td>
<td>5.17</td>
<td>591</td>
<td>5.17</td>
</tr>
<tr>
<td>Kingsoft Internet Security</td>
<td>6945</td>
<td>0.44</td>
<td>6945</td>
<td>0.44</td>
</tr>
<tr>
<td>McAfee VirusScan</td>
<td>700</td>
<td>4.36</td>
<td>700</td>
<td>4.36</td>
</tr>
<tr>
<td>Microsoft Forefront</td>
<td>841</td>
<td>3.63</td>
<td>841</td>
<td>3.63</td>
</tr>
<tr>
<td>Microsoft OneCare</td>
<td>1034</td>
<td>2.95</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Norton AntiVirus</td>
<td>618</td>
<td>4.94</td>
<td>618</td>
<td>4.94</td>
</tr>
<tr>
<td>Quick Heal AntiVirus</td>
<td>300</td>
<td>10.18</td>
<td>596</td>
<td>5.13</td>
</tr>
<tr>
<td>Symantec Endpoint Protection</td>
<td>430</td>
<td>7.10</td>
<td>452</td>
<td>6.76</td>
</tr>
<tr>
<td>VirusBuster Professional</td>
<td>462</td>
<td>6.61</td>
<td>2645</td>
<td>1.16</td>
</tr>
</tbody>
</table>

Kaspersky’s latest offering provoked considerable enthusiasm from our test engineer, who was impressed by the wide range of protective layers provided as well as the pleasant and informative interface with its range of data displays, including rolling graphs of monitored files and blocked attacks.

Glancing over the test results, scanning speeds were similarly impressive, and detection rates at their usual high level.
With the full range of VB100 requirements met without difficulty, Kaspersky also makes the grade and wins the award.

**Kingsoft Internet Security 2008.11.6.63**

<table>
<thead>
<tr>
<th>ITW</th>
<th>100.00%</th>
<th>Polymorphic</th>
<th>34.95%</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITW (o/a)</td>
<td>100.00%</td>
<td>Trojans</td>
<td>47.39%</td>
</tr>
<tr>
<td>Worms &amp; bots</td>
<td>99.27%</td>
<td>False positives</td>
<td>0</td>
</tr>
</tbody>
</table>

Kingsoft has been on a bit of a rollercoaster of late, with various product and detection issues meaning its record of VB100s has been somewhat sporadic. This time, however, the product seemed to behave itself for the most part.

After a rather lengthy but mostly quite straightforward installation process, testing ran along mainly using the default settings, as in-depth configuration was limited. No signs of the product’s previous instability issues were evident. The only problem encountered was in accessing the logs, as the various buttons to access the ‘log viewer’ system appeared disabled. After some poking around we found that this was another UAC problem, silent this time, and the logs could be accessed by running the viewer with admin rights from the start menu.

Speeds were remarkable, although not for the happiest of reasons, and detection rates left much to be desired in several sets. However, the WildList samples all detected correctly and no files were falsely alerted on in the clean test set, which means that Kingsoft makes the grade for another VB100 award.

**McAfee VirusScan Enterprise 8.7.0i**

<table>
<thead>
<tr>
<th>ITW</th>
<th>100.00%</th>
<th>Polymorphic</th>
<th>100.00%</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITW (o/a)</td>
<td>100.00%</td>
<td>Trojans</td>
<td>82.53%</td>
</tr>
<tr>
<td>Worms &amp; bots</td>
<td>100.00%</td>
<td>False positives</td>
<td>0</td>
</tr>
</tbody>
</table>

A quite different kettle of fish, McAfee’s product is a veteran war horse which has weathered many VB100s with barely a stumble. The product remains little changed, presenting a plain and unfussy face to the world but providing all the fine-tuning options expected of an enterprise-level product beneath its bonnet.

Our test engineer felt VirusScan was rather more affected by UAC blocks than some other products, and the system took considerably longer than usual to regain its desktop after the post-install reboot, but otherwise no issues were observed, with good scanning speeds and very good detection rates. No problems in the WildList or clean sets means that McAfee qualifies comfortably for another VB100 award.

**Microsoft Forefront Client Security 1.5.1955.0**

<table>
<thead>
<tr>
<th>ITW</th>
<th>100.00%</th>
<th>Polymorphic</th>
<th>96.04%</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITW (o/a)</td>
<td>100.00%</td>
<td>Trojans</td>
<td>85.14%</td>
</tr>
<tr>
<td>Worms &amp; bots</td>
<td>100.00%</td>
<td>False positives</td>
<td>0</td>
</tr>
</tbody>
</table>

Microsoft’s corporate product offers considerably fewer of those fine-tuning options than the product discussed above, at least at the desktop level, presenting an interface described by our test engineer as ‘very simple’, with not many options but lots of help. The absence of in-depth
configuration made for fairly straightforward testing, and the defaults seemed generally fairly sensible, with in-depth logging, not previously revealed to us by the developers, finally put to good use and the awkward event log system no longer required. One oddity of the logging was frequent warnings about ‘expensive’ files, but as there is nothing in the VB100 rules about overestimating the value of software, we let this pass.

The results showed fairly decent scanning speeds and detection rates were once again greatly improved. No false positives, and still no WildList misses, means another VB100 award goes to Forefront this month.

**Microsoft OneCare 2.5.2900.15**

- **ItW** 100.00% Polymorphic 96.04%
- **ItW (o/a)** 100.00% Trojans 81.16%
- **Worms & bots** 100.00% False positives 0

Many will have been shaken this month by the news that OneCare, Forefront’s home-user sibling, is to be retired next year and replaced by a livelier, simpler model. VB will not
be mourning too deeply, although we will be looking forward with some interest to the new version.

As it stands, OneCare is even simpler than Forefront, despite the various extra functions included, such as backup and disk defragmentation. There is very little opportunity for the user to manipulate its behaviour beyond the very basics, and on several occasions we found test systems completely crippled, and scan logs rendered inaccessible, by unexpected runs of ‘tune-up’ tasks. On most occasions, once this problem was diagnosed and the tasks aborted, a simple reboot allowed access to logs once more, despite the ominous tone of the error messages.

Speed results proved very slightly slower than Forefront, and detection rates pretty similar, with an identical lack of difficulties in the WildList and clean set resulting in another VB100 award for Microsoft.

**Norman Virus Control 5.99**

<table>
<thead>
<tr>
<th>Type</th>
<th>ItW</th>
<th>Polymorphic</th>
<th>ITW (o/a)</th>
<th>Polymorphic</th>
</tr>
</thead>
<tbody>
<tr>
<td>ItW</td>
<td>100.00%</td>
<td>81.03%</td>
<td>100.00%</td>
<td>63.65%</td>
</tr>
<tr>
<td>ITW (o/a)</td>
<td>100.00%</td>
<td>63.65%</td>
<td>100.00%</td>
<td>63.65%</td>
</tr>
<tr>
<td>Worms &amp; bots</td>
<td>100.00%</td>
<td>False positives 1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Norman’s unusual multi-interface approach made for more interference than usual from the UAC pop-ups, as various different parts of the product required individual confirmation. This meant that there were some rather long pauses moving from one part to another in the process of carrying out our various tasks. Configuration was patchy – in-depth in some areas and apparently absent in others, but things got done once the control system had been deciphered.

Scanning speeds were somewhat below average, but detection rates were pretty reasonable in general, with no difficulty covering the WildList. In the clean sets, however, a single file was mislabelled as the notorious Zlob trojan, and this was enough to spoil Norman’s chances for a VB100 award this month.

**Quick Heal AntiVirus 9.50**

<table>
<thead>
<tr>
<th>Type</th>
<th>ItW</th>
<th>Polymorphic</th>
<th>ITW (o/a)</th>
<th>Polymorphic</th>
</tr>
</thead>
<tbody>
<tr>
<td>ItW</td>
<td>100.00%</td>
<td>81.31%</td>
<td>100.00%</td>
<td>56.55%</td>
</tr>
<tr>
<td>ITW (o/a)</td>
<td>100.00%</td>
<td>56.55%</td>
<td>100.00%</td>
<td>56.55%</td>
</tr>
<tr>
<td>Worms &amp; bots</td>
<td>95.79%</td>
<td>False positives 0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Quick Heal proved once again to be worthy of its name, with testing completed fairly quickly once our test engineer had found his way around the interface. He described the interface as ‘bizarrely laid out’, and said that it seemed to keep some of its important functions quite well hidden.

Along with the excellent scanning speeds went less-than-superb detection rates, with detection for large numbers of items recently retired from the WildList apparently removed from databases – presumably to maintain that excellent scan rate. The WildList itself, however, was covered without problems, and without false positives Quick Heal is worthy of a VB100 award.
Rising Antivirus 20.67.10

ItW 100.00% Polymorphic 60.80%
ItW (o/a) 100.00% Trojans 53.98%
Worms & bots 99.80% False positives 0

Rising is another relative newcomer to the VB100 award, but the company has done pretty well so far with a nicely designed product. The setup process in this case was fairly complex, with a yellow warning from the UAC system and further configuration requirements after the reboot.

The stability that was noted with approval in previous tests was sadly less evident this time, with some oddities of behaviour and downright crashes slowing down the progress of our testing. On one occasion, after an on-demand scan of clean files, the ominous message ‘Rising Antivirus has stopped working’ appeared, while several times during on-access testing file accesses seemed to accelerate rapidly, and detections cut off completely, implying that the on-access scanner had also cut out.

After several retries some reasonably reliable results were obtained, showing some rather sluggish scanning speeds and less-than-perfect detection rates, but the WildList at least was well handled, and without false positives Rising also qualifies for a VB100 award.

Sophos Anti-Virus 7.6.1

ItW 100.00% Polymorphic 93.34%
ItW (o/a) 100.00% Trojans 71.37%
Worms & bots 100.00% False positives 0

Sophos is another veteran participant in the VB100, and the product impressed our tester with its speed of installation, well laid out interface and depth of configuration. UAC prompts seemed to accompany most selections from the main part of the interface.

Scanning speeds were pretty fast, at least with the default settings, and detection rates generally decent too, with no problems in the WildList and no false positives. Sophos thus joins the ranks of this month’s VB100 award winners.

Symantec Endpoint Protection 11.0.3001.2224

ItW 100.00% Polymorphic 70.63%
ItW (o/a) 100.00% Trojans 68.74%
Worms & bots 100.00% False positives 0

Symantec’s business desktop product has had a serious redesign of late, and the new look and feel has moved sharply away from the business-like, configurable simplicity of the previous edition towards the colourful and over-simplified. Though scanning speeds were decent, there were some very noticeable lags on various button presses, especially when trying to access the logs. These were most likely caused by the raw data, which in the case of our test runs often ran to hundreds of megabytes. Similar lags were also observed after some longer scan jobs.

Nevertheless, detection rates were solid, with no problems in the WildList or in the clean sets, and thus Symantec wins another VB100 award too.

VirusBuster Professional 6.0 build 206

ItW 100.00% Polymorphic 78.21%
ItW (o/a) 100.00% Trojans 53.16%
Worms & bots 99.94% False positives 0

VirusBuster’s installer needed to be run with full admin rights to function, but was still interrupted by the UAC prompts. After this, the installation was fast and simple, all done in less than 30 seconds, but this speed did not extend to the testing, with our tester finding the interface ‘appalling’. The convoluted layout, and lack of progress information on scanning times, didn’t make VirusBuster any new friends in the VB lab.

While on-demand scanning times were pretty decent, on-access times could not be gathered, as the product’s usual on-read detection appeared not to be functioning as expected on the platform under test. On-access results were thus obtained by copying file sets to the system with the product set to delete, and analysing the remains to measure accuracy. This proved mostly quite decent, with no difficulties in the WildList and no false positives, and VirusBuster is thus awarded another VB100 for its efforts.

Webroot I.S. Essentials 6.0.2.22

ItW 100.00% Polymorphic 89.83%
ItW (o/a) 100.00% Trojans 68.84%
Worms & bots 100.00% False positives 0

Webroot’s ‘WISE’ was another product that failed to impress our tester, with an interface that looked attractive on the surface, but quickly grew ugly when trying to do anything beyond the very basics. Configuration is very minimal, and
responsiveness somehow even lower, with huge time lags between various components, most notably the file browsing to select areas to scan. Scanning times were also rather slow, and although once again the on-access component appeared not to be sparked by simple file accesses, times were still recorded for this test as they were in some cases slower than other products that did scan the files.

Logging also proved an issue, with all data discarded after 1,000 lines, not much by the standards we require. Nevertheless, with a combination of careful scanning of tiny portions of our sets at a time, copying files around and to the system and allowing the product to mangle them as it saw fit, then comparing the results with the originals to check for changes, we finally managed to get some usable results. The results seemed to tally fairly closely with those of the Sophos engine at the core of the product’s detection capabilities. This meant that there were no issues in either the WildList or clean sets, and that, despite annoying the test team quite thoroughly, Webroot earned another VB100 award.

**CONCLUSIONS**

On top of the already rather arduous task of getting through multiple tests on multiple products, this month presented more than the average number of small annoyances and petty frustrations. These included bizarre and buggy interfaces, hidden or absent options, and unreliable behaviours, as well as a few more major issues, including product freezes and crashes, blatant contradiction of advertised behaviours, and the occasional product which all but defied testing. Much of this can be put down to the less than ubiquitous platform, but developers claiming to support a given platform need to ensure that their products undergo full and thorough quality assurance. This month’s test has been notable for the more than usually high number of passes – indeed only two products failed to meet the required standard, with a couple of other close calls. It seems appropriate to remind readers that we expect products to pass our base test requirements on a regular basis, that the VB100 requirements are not intended as an indication of superlative products, merely of adequate and reasonably reliable ones. The purpose of the scheme is to provide certification of products proven to be legitimate, and to provide a basic level of protection. A single test result should not be taken in a vacuum, but patterns and trends of performance over time can be a valuable guide to the trustworthiness of a product and its developers.

In addition to the plain pass/fail outcomes which some take to be the be-all and end-all of the VB100, we provide a wide range of additional information on the products that take part, including measurements of scanning speed and overheads, overviews of additional functionality and usability, and detection rates over a selection of additional test sets.

In the next VB100, we plan to introduce a major addition to this range of extras, based on weekly test sets built up in the weeks immediately prior to, and shortly after, the product submission deadline. This should provide a useful indicator of how well developers are keeping up with the ever-growing flood of new malware samples seen on a daily basis, many of them frequently morphed and fine-tuned with the explicit aim of avoiding detection by anti-malware software. The test should also provide some insight into how well heuristic and generic detection techniques are allowing products to detect malware as yet unseen by analysis labs.

Details of the new test system, which we have dubbed ‘RAP’, standing for Reactive And Proactive measuring, were presented at the recent VB conference and have since been opened to deeper consultation with interested parties. In its fully developed form the new test should provide clear and easily understood additional data, which will also build up over time to show long-term trends and patterns of improvement, stagnation or decline in the performance of scanner products.

This will go some way to providing more information on the performance and capabilities of current security software, but of course there are many diverse new functions being added to solutions with every new generation, many of which require major shifts in test design to properly measure their efficacy. In conjunction with groups like AMTSO, our own expert advisory board and other interested parties, we will continue to investigate and develop new testing methodologies, and even new certification schemes, that will enable us to more accurately evaluate the products’ full capabilities. We hope to make many more strides in this direction in the course of the coming year, and as always we welcome any feedback, input, suggestions and opinions from our readers.
END NOTES & NEWS

The 2nd Annual Chief Security Officer Summit will take place 8–10 December 2008 in Geneva, Switzerland. The summit aims to bring together security directors from across Europe, Africa and the Middle East to tackle the most critical and strategic security challenges at the highest business level. For more information see http://www.mistieurope.com/csos/.

ACSC 24 (the Applied Computer Security Associates’ Annual Computer Security Conference) will be held 8–12 December 2008 in Anaheim, CA, USA. For details see http://www.acsac.org/.

AVAR 2008 will be held 10–12 December 2008 in New Delhi, India. The 11th Association of anti-Virus Asia Researchers International Conference will be hosted by Quick Heal Technologies Pvt. See http://www.avar.org/avar2008/index.htm.

Black Hat DC 2009 takes place 16–19 February 2009 in Washington, DC, USA. Online registration is now open and a call for papers has been issued (deadline 1 January 2009). For details see http://www.blackhat.com/.

CanSecWest 2009 will take place 16–20 March 2009 in Vancouver, Canada. Those interested in presenting at the event should submit proposals by 8 December. For full details see http://cansecwest.com/.

The 3rd Annual Securasia Congress takes place in Kuala Lumpur, Malaysia, 25–26 March 2009. Key topics include global threats to security, social engineering and malware trends, addressing the insider threat to database security and developing meaningful security metrics for security management. For full details see http://www.securasia-congress.com/.

Black Hat Europe 2009 takes place 14–17 April 2009 in Amsterdam, the Netherlands, with training taking place 14–15 April and the briefings part of the event from 16–17 April. Registration is now open and a call for papers has been issued (deadline 1 February 2009). See http://www.blackhat.com/.

RSA Conference 2009 will take place 20–24 April 2009 in San Francisco, CA, USA. The conference theme is the influence of Edgar Allen Poe, a poet, writer and literary critic who was fascinated by cryptography. For more information including registration rates and packages see http://www.rsaconference.com/2009/us/.

Infosecurity Europe 2009 takes place 28–30 April 2009 in London, UK. For more details see http://www.infosec.co.uk/.

The 18th USENIX Security Symposium will take place 12–14 August 2009 in Montreal, Canada. For more information see http://www.usenix.org/events/sec09/.

VB2009 will take place 23–25 September 2009 in Geneva, Switzerland. For details of sponsorship opportunities and any other queries relating to VB2009, please email conference@virusbtn.com.

ADVISORY BOARD
Pavel Baudis, Alwil Software, Czech Republic
Dr Sarah Gordon, Independent research scientist, USA
John Graham-Cumming, France
Shimon Gruper, Aladdin Knowledge Systems Ltd, Israel
Dmitry Gryaznov, McAfee, USA
Joe Hartmann, Microsoft, USA
Dr Jan Hruška, Sophos, UK
Jeannette Jarvis, Microsoft, USA
Jakub Kaminski, Microsoft, Australia
Eugene Kaspersky, Kaspersky Lab, Russia
Jimmy Kuo, Microsoft, USA
Anne Mitchell, Institute for Spam & Internet Public Policy, USA
Costin Raiu, Kaspersky Lab, Russia
Péter Ször, Symantec, USA
Roger Thompson, AVG, USA
Joseph Wells, Lavasoft USA

SUBSCRIPTION RATES
Subscription price for 1 year (12 issues):
- Single user: $175
- Corporate (turnover < $10 million): $500
- Corporate (turnover > $10 million): $1,000
- Corporate (turnover > $100 million): $2,000
- Bona fide charities and educational institutions: $175
- Public libraries and government organizations: $500

Corporate rates include a licence for intranet publication. See http://www.virusbtn.com/virusbulletin/subscriptions/ for subscription terms and conditions.

Tel: +44 (0)1235 555139 Fax: +44 (0)1865 543153
Email: editorial@virusbtn.com Web: http://www.virusbtn.com/

No responsibility is assumed by the Publisher for any injury and/or damage to persons or property as a matter of products liability, negligence or otherwise, or from any use or operation of any methods, products, instructions or ideas contained in the material herein.

This publication has been registered with the Copyright Clearance Centre Ltd. Consent is given for copying of articles for personal or internal use, or for personal use of specific clients. The consent is given on the condition that the copier pays through the Centre the per-copy fee stated below.

Tel: +44 (0)1235 555139 /2008/$0.00+2.50.
No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form without the prior written permission of the publishers.
FEATURE

FIGHTING PHISHING AT THE BROWSER LEVEL

Alexandru C. Cosoi
BitDefender, Romania

Phishing can no longer be considered a new and emerging phenomenon. Fake websites impersonating both national and international financial institutions appear everywhere, trying to manipulate users into giving away their credentials. This article describes a method that attempts to deal with the phishing problem at the browser level, by combining both whitelisting and content-based solutions into a web page forgery detector.

INTRODUCTION

Phishing is a form of social engineering in which an attacker attempts to acquire sensitive information from a victim by impersonating a trustworthy third party.

In a typical phishing attempt, a fake website (also termed a clone) poses as a genuine web page belonging to an online retailer or a financial institution, and the user is asked to enter some personal information (e.g. username and password) and/or financial information (e.g. credit card number, bank account number, security code etc.). Once the information has been submitted by the unsuspecting user, it is harvested by the attacker. The user may also be directed to a web page which installs malicious software (e.g. viruses, trojans) on the user’s computer. The malicious software may continue to steal personal information by recording the user’s keystrokes while visiting certain web pages, and may transform the user’s computer into a platform for launching other phishing or spam attacks.

Current anti-spam technologies have achieved competitive detection rates against phishing emails, but recently phishers have started using a number of different mediums to lure users to their fake websites – including instant messaging, social networks, blog posts and even SMS [1, 2]. Once attackers have some basic information about their victims from social network profiles [3], it is easy for them to socially engineer their way into the users’ trust. This makes it even more important for browser-level protection to prevent the user from accessing the malicious websites.

NEWS & EVENTS

SPAM LEVELS BOUNCE BACK

Spam levels have started to rise again just two weeks after a massive drop when web-hosting firm McColo was taken offline.

McColo – which hosted botnet control centres that controlled zombies around the world and which were responsible for more than 75% of the spam sent globally each day – was taken offline by its upstream providers last month after security researcher Brian Kreb presented them with evidence about suspicious activities on the provider’s network. Spam levels plummeted almost instantly.

Now, however, spam levels have begun to rise again. Estimates vary as to the extent of the rise, IronPort Systems putting the level of spam at less than half that prior to the McColo shutdown, while MessageLabs believes the level to have risen to around two-thirds that prior to the shutdown. Of course, none of this comes as a surprise – researchers have been expecting to see botnets kick back into action and the last few days of November saw the resurrection of the Rustock and Srizbi botnets, each of which is capable of sending massive amounts of spam.

While levels are currently lower than prior to the McColo shutdown, there is little doubt that this month will see spam levels reach record heights again.

EVENTS

The 15th general meeting of the Messaging Anti-Abuse Working Group (MAAWG) will be held in San Francisco, CA, USA, 17–19 February 2009. The 16th and 17th general meetings will be held 9–11 June 2009 in Amsterdam, The Netherlands, and 27–29 October 2009 in Philadelphia, PA, USA, respectively. For full details see http://www.maawg.org/.

The Counter-eCrime Operations Summit will be held 12–14 May 2009 in Barcelona. For more details see http://www.antiphishing.org/.
Current browser-based technologies employ whitelisting and blacklisting techniques, various heuristics to see if URLs are similar to well-known legitimate URLs, community ratings and content-based heuristics [4], and lately visual similarity techniques [5].

Blacklisting has worked well so far, but the time it takes for a URL to become blacklisted worldwide overlaps in most cases with the time frame in which the phishing attack is most successful. Also, not all of the current content-based solutions make use of whitelists, which can result in the misclassification of sites – for example a filter might treat the official eBay website as a phishing site [6].

**PROPOSED METHOD**

In developing our method we started from the following hypothesis: in a given language, the number of different possible ways of phrasing a message that transmits the same or similar information (such as ‘Please log into your online banking account in order to access your funds’) is limited by the writer’s common sense (i.e. the information must be phrased in a simple, readable and understandable form). In other words, we assume that the English login pages of financial institutions will have a large set of common words, since they share a common purpose and specialized financial vocabulary [7–12].

Two documents, A and B (in our case the web pages of financial institutions such as PayPal or Bank of America), can be represented as sets of words:

\[ A = C \cup N_1 \]  
\[ B = C \cup N_2 \]

where C represents the common words between the two documents, and \( N_1 \) and \( N_2 \) the distinct words. This means that the number of words needed to construct a database with triples of the form (word, document, occurrences), is:

\[ |C| + |N_1| + |N_2| \leq 2|C| + |N_1| + |N_2| \]

or in short  \( |A \cap B| \leq |A \cup B| \). In the case of only two documents, this technique might not be very useful, but in the case of several documents which serve the same purpose (e.g. the websites of financial institutions), we can assume that the outcome will consist of a large number of common words.

We will now define a similarity indicator between two documents, known as the Jaccard distance\(^1\) for sets:

\[ d = 1 - \frac{|A \cap B|}{|A \cup B|} \tag{1} \]

On identical documents, this distance will have a null value, while in the case of similar, but non-identical documents it will be close to 0. Since these are not standard sets (e.g. in ordinary sets, identical elements appear just once, while in this set, we decided that each element – or word – appears as many times as it is found in the document), the distance actually provides an acceptable similarity value, based on the number of words.

On a corpus of 101 financial institutions from three different countries – the top five phished banks in Romania, seven websites from Germany and 89 randomly chosen US institutions which showed a high frequency of email phishing in our internal email corpus, with an average of 100 words per page – we obtained a database of just 4,422 different words, instead of an expected minimum of 10,000 words.

Considering a pool of web pages (such as those described above), we can construct a database in the format presented in Table 1.

<table>
<thead>
<tr>
<th></th>
<th>Bank 1</th>
<th>Bank 2</th>
<th>Bank 3</th>
<th>Bank 4</th>
<th>Bank 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Word 1</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Word 2</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 1: Database format.

Based on this initial background, our proposed method is outlined in Figure 1. First, the presented web page is verified against a blacklist and a whitelist. Afterwards, some simple heuristics are run on the content of the web page, to check whether it tries to mimic an official login page.
(e.g. contains a submit button or key words such as eBay, PayPal, etc.). We introduced this step for speed optimization purposes (it would be pointless to check if a web page with no submit form tries to duplicate a web page that has such a form).

If we consider it necessary to run our forgery filter on a target web page, then we start computing the Jaccard distance for each institution on which the filter has trained (the words from learned web pages are stored in the database). The lowest distance obtained indicates the highest similarity between the target web page and a reference web page in our database. If the computed distance is smaller than a predefined threshold, we consider the target website to be forged.

When dealing with this technology, the use of an up-to-date whitelist is essential in order to prevent the forgery filter being run on original websites, and thus prevent false positives.

RESULTS

Usually, if the filter has been trained on a certain web page, we will have a similarity distance of at least 0.01, and experimentally (using over 10,000 samples) we never obtained a distance of higher than 0.2 on phishing websites. For training, we used a corpus of 101 pages (presented earlier) and a value of 0.25 for the similarity threshold.

We tested our filter on two different corpuses: one containing 10,000 forged websites of the exact pages on which the filter has trained (randomly selected from real phishing pages) and the other containing the URLs published on PhishTank over a period of 10 days.

We obtained a 99.8% detection on the first corpus, with 20 false negatives – which were mostly due to the fact that they were generated with screenshots from the original web page and did not show enough text content for a discriminative decision. We obtained a 42.8% detection on the PhishTank URLs. Although this may seem low, our data indicates that we obtained these results due to the fact that 144 hijacked brands were co-opted in phishing attacks in December 2007 – which was far more than in our training corpus.

This experiment can easily be reproduced if, in a multicategorical Bayesian filter, we swap the probability function with equation 1 and the probability of each word belonging to a category will represent the number of occurrences of that word in that category. Then, if instead of choosing the category with the highest probability, we chose the category with the smallest distance, we would obtain the same results as presented above.

As for false positives, on a corpus of 25,000 samples of web pages containing login forms, or any other information that would activate the forgery filter, we obtained 10 false alarms. Eight of them were real financial institutions, which would have been in the whitelist if the filter had been properly trained, while the other two were genuine false positives (two online financial newspapers) and this problem can easily be solved by whitelisting the sites.

FUTURE DEVELOPMENTS

Right now, because of the lack of content-based solutions [4, 6], phishers are putting only a small amount of effort into customizing their forged websites (e.g. using random invisible content, frames, HTML obfuscation) and concentrating instead on rapidly changing their hosting addresses. Since initially we were expecting a higher rate of false positives, we also developed another distance:

\[ d = 1 - \frac{|A \Delta B|}{|A \cup B|} \]  

where \( A \) = \{ \( w \in A \cap B | \alpha - \epsilon \leq x_w - y_w \leq \alpha + \epsilon \) \}, wherein \( w \) represents a word, \( A \) represents the target word set, \( x_w \) represents a position index of the word \( w \) within the reference word set and \( y_w \) represents a position index of the word \( w \) within the reference word set. We accept a certain number of missing or extra words between different paragraphs if their number is between \( (\alpha - \epsilon) \), \( (\alpha + \epsilon) \) \) (e.g. we accept a variable but controlled difference between the target and the reference position index).

In equation 2 (which is in fact a modified Jaccard distance), \( d \) is close to 0 if the target word set and the reference word set share a large number of words in the same order of appearance, and \( d \) is close to 1 if the word sets have few common words and/or the words appear in a different order in the target and reference web pages.

We observed that, although the second distance provides a greater protection against false positives, it will also score more false negatives, since phishers sometimes change the order of phrases when forging a website.

CONCLUSIONS

Since phishing websites are no longer solely advertised through email, we believe that it is time for companies to invest more in the research and development of browser-level anti-phishing protection.

---

2 Only visible words will be inserted in the database.
3 http://www.phishtank.com/
5 A word set represents the list of all words found in the documents. If a word is found 10 times in a certain document, it will also be found 10 times in the word set.
The proposed method is intended to be used alongside current technologies, providing the user with extra information about visited web pages. Although not a complete solution on its own (it is ineffective on phishing websites that do not mimic the original website), when used in combination with other technologies (e.g. blacklists, content and URL heuristics) it increases the value of any anti-phishing toolbar.

The obtained results show that this is a viable method to provide forgery detection for the websites of legitimate financial institutions. It is not necessary to run this system on all the pages visited by the user, focusing just on those that require the user to submit information, thereby highly increasing the user’s tolerance level by decreasing the time required for analysis.

**ACKNOWLEDGEMENTS**

This work was supported by BitDefender AntiSpam Laboratory. The author thanks Mr Lucian Lupesescu and Mr Razvan Visan for their help in developing this project.

**REFERENCES & FURTHER READING**


