JANUARY 2009

Virus Bulletin

Fighting malware and spam

CONTENTS

2 **COMMENT**

Welcome to 2009

3 **NEWS**

Addendum: VB100 comparative review

Gates closed at CastleCops

Legal hacking debate gathers pace

3 **VIRUS PREVALENCE TABLE**

4 **TECHNICAL FEATURE**

Anti-unpacker tricks – part two

FEATURES

10 A day in the life of an average user

14 Advancing malware techniques 2008

16 **PRODUCT REVIEW**

MicroWorld eScan Internet Security Suite 10

21 **END NOTES & NEWS**

IN THIS ISSUE

CRYSTAL BALL

What will 2009 bring for computer security? Will we have a better year than the last one? Graham Cluley makes some predictions.

page 2

DANGER ZONE

Are we being attacked every minute? Gabor Szappanos researches the dangers of being an average computer user.

page 10

ESCAN ON TEST

The VB testing team put MicroWorld Technologies' eScan Internet Security Suite through its paces and find a top-class level of protection with much to offer.

page 16

Protection

eScan Protection Center provides Total Digital Immunity from viruses, spyware, adware, rootkits and other malicious programs.

✓ File Anti-Virus
✓ Mail Anti-Virus
✗ Anti-Spam
✓ Web Protection
✓ Firewall
✓ Endpoint Security
✗ Privacy Control

Date of virus signature: 13 Dec 2008 13:55

Last computer scan: Not yet scanned

vbSpam supplement

This month: anti-spam news and events, and Martijn Grooten outlines the proposed test set-up for Virus Bulletin's upcoming anti-spam comparative testing.

ISSN 1749-7027

virus BULLETIN COMMENT

'[In 2009] the web will be infected like never before.'

Graham Cluley
Sophos, UK

WELCOME TO 2009

If you're anything like me you'll have risen on 1 January 2008 full of optimism that it would be a better year for computer security. But, let's be honest, it wasn't that great was it?

We saw more malware being produced than ever before, as the bad guys sped up their conveyor belts creating Trojan horses and injected malicious code into websites at a much faster rate than ever before.

Well-known brand names like *BusinessWeek*, *Sony* and *Adobe* were amongst those who fell foul of SQL injection attacks, suffering the humiliation of infecting visiting customers because of sloppy website coding.

Scareware (or fake anti-virus products) also barnstormed into prominence like never before. The security community has been trying to raise awareness of computer security amongst the general public for years, but ironically the advice many listened to was from scammers trying to fool them into buying bogus products with fake alert messages. Hacking gangs have become proficient at producing professional-looking websites posing as legitimate security vendors.

2008 saw a surge in both spammers and malware authors turning to social networks like *Facebook* – stealing the usernames and passwords of the unwary in their attempts to sell dubious wares or find new computers to infect.

With this and other security threats causing headaches for computer users, it's not surprising that many of us are thinking 'good riddance to 2008'.

But what will 2009 bring? Will we have a better year than the last one? Well, here are some of my predictions.

Editor: Helen Martin

Technical Consultant: John Hawes

Technical Editor: Morton Swimmer

Consulting Editors:

Nick FitzGerald, *Independent consultant, NZ*

Ian Whalley, *IBM Research, USA*

Richard Ford, *Florida Institute of Technology, USA*

First of all, the easy ones, as some things do seem certain. For instance, the variety of attacks and their number will continue to escalate. Modern-day malware is driven by the desire of organized criminal gangs to break into computers to steal information, identities and resources.

Botnets will continue to be a key component of cybercrime. Compromised PCs, both at home and at work, will still be the primary source of spam in 2009. McColo's disconnection from the net was a victory that should be celebrated as it brought down botnet command-and-control centres, but its impact on spam levels will be short-lived. More botnets in future will adopt a decentralized, P2P-style of operation, making quick wins harder to achieve.

The web will be infected like never before. While most computer users have their email scanned for malware, far fewer people are properly scanning the web content that is being sent to their desktops.

Websites are becoming more complex – you no longer have a static set of web pages that you can simply scan for malicious content, since pages are often constructed in real time, possibly from individual snippets of data from many different fields in many different tables in many different databases. Meanwhile, on the client side, browsers are becoming more complex. You need to rely on your browser – and perhaps a large number of add-on DLLs and plug-ins – to protect you from malicious or dangerous actions programmed into scripts.

I predict we will see more use of non-EXE files by criminals in 2009. We can expect to be fighting legitimate-looking data files, such as *Word* DOCs and PDFs, that are booby-trapped with exploits against software vulnerabilities.

Finally, a prediction which may not come true in 2009, but surely soon will. Towards the end of last year there was an almighty kerfuffle as *Apple* pushed out conflicting messages regarding the need or otherwise for anti-virus software on the *Mac* platform. Whatever the scale of the malware problem on *Apple Mac*s at the moment, it seems inevitable that the cybercriminals will find it irresistible to launch more attacks against a community which has largely taken a laissez-faire attitude to security.

Sounds gloomy doesn't it? But I believe that, if managed properly, the problem should not be insurmountable. The good news is that security vendors are improving all the time, and sharing their expertise via industry initiatives like the *Virus Bulletin* conference. Proactive detection of new, unknown malware threats is at an all-time high, and computer users can dramatically reduce the risks in the year ahead by following sound security practices and using up-to-date protection.

NEWS

ADDENDUM: VB100 COMPARATIVE REVIEW

VB regrets that, due to an oversight, the *MicroWorld Technologies eScan* product was omitted from the write-up of the VB100 comparative review on *Windows Vista x64* (see *VB*, December 2008, p.14). In fact, *eScan* detected all of the samples in the WildList test set and did not generate any false positives when scanning the clean test set – thus the product qualifies for a VB100 award. *VB* extends its apologies to *MicroWorld* for the omission.

GATES CLOSED AT CASTLECOPS

After nearly seven years of fighting cybercrime, volunteer-run and much respected security operation *CastleCops* closed its doors last month.

Established in 2002, the work undertaken by *CastleCops* included the investigation of malware and phishing scams, malicious site take-downs and security training programs. The organization forged close ties both with the anti-malware community and with law enforcement agencies. *CastleCops* suffered its share of denial of service attacks and defamation attempts over its close-to-seven-year history, but it seems likely that the lack of a new leader following *Microsoft*'s hiring of founder Paul Laudanski in June 2008 was one of the contributing factors to the closure of the group.

An announcement on the *CastleCops* website reads: 'It has been our pleasure to investigate online crime and volunteer with our virtual family to assist with your computer needs and make the Internet a safer place. Unfortunately, all things come to an end.'

LEGAL HACKING DEBATE GATHERS PACE

The argument over whether law enforcement agencies should use spyware to keep an eye on suspected criminals – which has been rumbling around for many years (see *VB*, April 2007, p.2) – has gathered new pace as reports in the UK press suggest that British police have been given the power to hack into computers without a court warrant. While there are obvious advantages to using spyware and keylogging devices to observe criminal activity and gather evidence, the idea of doing so without case-by-case review and approval from a court raises serious questions over civil liberties. There are also concerns over the security of such practices – for example the risk of such programs spreading in the wild or being hacked – and it is very unlikely that the AV industry would ever agree to the non-detection of such programs. Writing in his blog this week, *Sophos* spokesman Graham Cluley said 'we will continue to defend computer users against malware and spyware, regardless of who might have written or installed the code. And if that puts us at loggerheads with our friends in the police, so be it.'

Prevalence Table – November 2008

Malware	Type	%
Agent	Trojan	27.66%
Invoice	Trojan	19.06%
Goldun	Trojan	9.93%
NetSky	Worm	6.31%
Autorun	Worm	6.03%
Zbot	Trojan	4.40%
Mytob	Worm	3.93%
Virut	Virus	3.35%
Dropper-misc	Trojan	3.08%
Basine	Trojan	2.84%
Mydoom	Worm	2.56%
Suspect packers	Misc	2.55%
Bagle	Worm	1.02%
Small	Trojan	0.86%
Iframe	Exploit	0.57%
Delf	Trojan	0.49%
Rootkit-misc	Trojan	0.48%
Murlo	Trojan	0.47%
Downloader-misc	Trojan	0.46%
Hupigon	Trojan	0.44%
Zafi	Worm	0.42%
Grew	Worm	0.39%
Heuristic/generic	Misc	0.37%
OnlineGames	Trojan	0.34%
Zlob/Tibs	Trojan	0.26%
Sality	Virus	0.23%
Inject	Trojan	0.22%
PWS-misc	Trojan	0.21%
Lineage/Magania	Trojan	0.18%
Klez	Worm	0.11%
Womble	Worm	0.09%
Heuristic/generic	Trojan	0.08%
Cutwail/Pandex/Pushdo	Trojan	0.07%
Others[1]		0.54%
Total		**100.00%**

[1]Readers are reminded that a complete listing is posted at http://www.virusbtn.com/Prevalence/.

TECHNICAL FEATURE

ANTI-UNPACKER TRICKS – PART TWO

Peter Ferrie
Microsoft, USA

In the first part of this series last month (see *VB*, December 2008, p.4) we looked at a number of anti-unpacking tricks that have come to light recently. New anti-unpacking tricks continue to be developed because the older ones are constantly being defeated. In this article and the ones that follow, we will describe some tricks that might become common in the future, along with some countermeasures.

INTRODUCTION

Anti-unpacking tricks come in different forms, depending on what kind of unpacker they are intended to attack. The unpacker can be in the form of a memory-dumper, a debugger, an emulator, a code-buffer, or a W-X interceptor. It can also be a tool in a virtual machine. There are corresponding tricks for each of these.

- 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.
- A *code-buffer* is similar to a debugger. It also attaches to a process, but instead of executing instructions in place, it copies each instruction into a private buffer and executes it from there. It allows fine-grained control over execution as a result. It is also more transparent than a debugger, and faster than an emulator.
- A *W-X interceptor* uses page-level tricks to watch for write-then-execute sequences. Typically, an executable region is marked as read-only and executable, and then everything else is marked as read-only and non-executable (or simply non-present, depending on the hardware capabilities). Then the code is allowed to execute freely. The interceptor intercepts exceptions that are triggered by writes to read-only pages, or execution from non-executable or non-present pages. If the hardware supports it, a read-only page will be replaced by a writable but non-executable page, and then the write will be allowed to continue. Otherwise, the single-step exception will be used to allow the write to complete, after which the page will be restored to its non-present state. In either case, the page address is kept in a list. In the event of exceptions triggered by execution of non-executable or non-present pages, the page address is compared to the entries in that list. A match indicates the execution of newly written code, and is a possible host entry point.

Now we move to potentially new tricks. All of these techniques were discovered and developed by the author of this paper. This article will concentrate on anti-debugging tricks.

ANTI-UNPACKING BY ANTI-DEBUGGING

1. Heap flags

Within the heap are two fields of interest. The PEB->NtGlobalFlag field forms the basis for the values in those fields. It should be noted that the HEAP_VALIDATE_PARAMETERS_ENABLED flag value was changed in *Windows XP* and later, from 0x200000 to 0x40000000, and that a new NtGlobalFlag flag 0x80 (FLG_HEAP_VALIDATE_ALL) was introduced (which corresponds to the HEAP_VALIDATE_ALL_ENABLED flag). Further, the location of the Flags and ForceFlags fields is different in *Windows Vista*. No current packer supports the new location, which is the reason why some packers will not run on *Windows Vista*.

Example code for *Windows Vista* looks like this:

```
mov eax, fs:[30h] ;PEB
;get process heap base
mov eax, [eax+18h]
mov eax, [eax+40h] ;Flags
dec eax
dec eax
jne being_debugged
```

and this:

```
mov eax, fs:[30h] ;PEB
;get process heap base
mov eax, [eax+18h]
cmp d [eax+44h], 0 ;ForceFlags
jne being_debugged
```

2. Special APIs

2.1 CreateFile

The kernel32 CreateFile() function can be used to open a file for exclusive access. This technique is not new in general, but it is new with respect to debugger detection techniques.

Example code looks like this:

```
    xor   ebx, ebx
    mov   ebp, offset l1
    push  104h ;MAX_PATH
    push  ebp
    push  ebx ;self filename
    call  GetModuleFileNameA
    push  ebx
    push  ebx
    push  3 ;OPEN_EXISTING
    push  ebx
    push  ebx
    push  80000000h ;GENERIC_READ
    push  ebp
    call  CreateFileA
    inc   eax
    je    being_debugged
    ...
l1: db    104h dup (?) ;MAX_PATH
```

This technique works against the debugger *Turbo Debug32*, but not debuggers such as *OllyDbg* and *WinDbg*. It is related to the debug privilege, which debuggers such as *OllyDbg* and *WinDbg* maintain, while *Turbo Debug32* does not.

2.2 RaiseException

The kernel32 RaiseException() function can be used to force certain exceptions to occur. These include exceptions that a debugger would normally consume.

Turbo Debug32 consumes the following exceptions:

```
0x40010005 (DBG_CONTROL_C)
0x40010007 (DBG_RIPEVENT)
0x80000002 (DATATYPE_MISALIGNMENT)
0x80000003 (BREAKPOINT)
0x80000004 (SINGLE_STEP)
0x80000029 (UNWIND_CONSOLIDATE)
0xC0000005 (ACCESS_VIOLATION)
0xC000008C (ARRAY_BOUNDS_EXCEEDED)
0xC000008D (FLOAT_DENORMAL_OPERAND)
0xC000008E (FLOAT_DIVIDE_BY_ZERO)
0xC000008F (FLOAT_INEXACT_RESULT)
0xC0000090 (FLOAT_INVALID_OPER)
0xC0000091 (FLOAT_OVERFLOW)
0xC0000092 (FLOAT_STACK_CHECK)
0xC0000093 (FLOAT_UNDERFLOW)
0xC0000094 (INTEGER_DIVIDE_BY_ZERO)
0xC0000095 (INTEGER_OVERFLOW)
0xC0000096 (PRIVILEGED_INSTRUCTION)
```

When raised in the presence of *Turbo Debug32,* none of these exceptions will be delivered to the debuggee. The missing exception can be used to infer the presence of *Turbo Debug32*.

Example code looks like this:

```
    xor  eax, eax
    push offset l1
    push d fs:[eax]
    mov  fs:[eax], esp
    push eax
    push eax
    push eax
    ;DBG_CONTROL_C
    push 40010005h
    call RaiseException
    jmp  being_debugged
l1: ...
```

By default, *OllyDbg* will consume a similar list of exceptions, but it can be configured to pass them to the debuggee.

The *Interactive DisAssembler* (*IDA*) debugger consumes the following exceptions:

```
0x40010006 (DBG_PRINTEXCEPTION_C)
0x40010007 (DBG_RIPEVENT)
0x80000003 (BREAKPOINT)
```

It is known that *WinDbg* consumes the DBG_PRINTEXCEPTION_C (0x40010006) exception, though this fact is used only rarely. However, *WinDbg* also consumes the following exceptions:

```
0x40000005 (SEGMENT_NOTIFICATION)
0x40010005 (DBG_CONTROL_C)
0x40010007 (DBG_RIPEVENT)
0x40010008 (DBG_CONTROL_BREAK)
0x40010009 (DBG_COMMAND_EXCEPTION)
0x80000001 (GUARD_PAGE_VIOLATION)
0xC0000420 (ASSERTION_FAILURE)
```

The SEGMENT_NOTIFICATION (0x40000005) exception is of particular interest, since it can be used to demonstrate several behaviours. One of these behaviours is to force a break into the VDM debugger prompt.

Example code looks like this:

```
    push offset l1
    push 4
    push 0
    ;EXCEPTION_SEGMENT_NOTIFICATION
    push 40000005h
    call RaiseException
    ...
l1: dd  0c0000002h, 0
    dd  offset l1, offset l1
    dd  0, 0, offset l1
    db  2b0h dup (0)
```

Another of the behaviours is to cause the debugger to remove a breakpoint from the specified location in the debuggee's process memory.

Example code looks like this:

```
    push offset l4
    push 4
    push 0
    ;EXCEPTION_SEGMENT_NOTIFICATION
    push 40000005h
    call RaiseException
    push offset l5
    push 1
```

```
      push 0
      ;EXCEPTION_SEGMENT_NOTIFICATION
      push 40000005h
      ;remove breakpoint
      call RaiseException
  l1: mov al, 0cch
      ...
  l2: dd  0
  l3: dd  offset l7
  l4: dd  2 ;dummy context request
  l5: dd  6, offset l2, offset l3
      dd  0, offset l2
  l6: db  3, 90h ;replacement value
      db  0ah dup (0)
  l7: dw  0
      db  offset l1 + 1
      db  (offset l1 + 1) shr 8
      db  (offset l1 + 1) shr 10h
      dw  0
      db  (offset l1 + 1) shr 18h
      dd  0, offset l6
      db  7ch dup (0)
      dw  1
      db  8 dup (0), 1, 209h dup (0)
```

In this case, the value in AL at l1 is altered from 0xCC to 0x90.

In *Windows Vista*, there are two new exceptions. They are EXCEPTION_WX86_SINGLE_STEP (0x4000001E) and EXCEPTION_WX86_BREAKPOINT (0x4000001F). As their names imply, they are the x86 equivalents of EXCEPTION_BREAKPOINT (0x80000003) and EXCEPTION_SINGLE_STEP (0x80000004). When a single-step or breakpoint occurs in 32-bit mode, these new exceptions are raised instead of the old ones. If a debugger does not handle them, then the kernel translates them to the old values and dispatches them again. In either case, they will be consumed by the debugger if that was the previous behaviour.

2.3 DbgBreakPoint

The ntdll DbgBreakPoint() function is called when a debugger attaches to a process that is already running. This allows the debugger to gain control because an exception is raised that it can intercept. This technique can be defeated simply by erasing the breakpoint.

Example code looks like this:

```
      push offset l1
      call GetModuleHandleA
      push offset l2
      push eax
      call GetProcAddress
      push eax
      push esp
      push 40h     ;PAGE_EXECUTE_READWRITE
      push 1
      push eax
      xchg ebx, eax
      call VirtualProtect
      mov  byte ptr [ebx], 0c3h
      ...
  l1: db   "ntdll", 0
  l2: db   "DbgBreakPoint", 0
```

If a debugger attempts to attach to a process that contains such a change, then the thread will exit immediately, and the debugger will not break in. *Turbo Debug32*, and possibly other console-mode debuggers, will hang as a result, because they wait infinitely for an exception to be raised in order to continue execution.

2.4 OutputDebugString

Despite the fact that the kernel32 OutputDebugString() function raises the DBG_PRINTEXCEPTION_C (0x40010006) exception, a registered Structured Exception Handler will not see it. The reason is that *Windows* registers its own Structured Exception Handler internally, which consumes the exception if a debugger does not do so. As such, the presence of a debugger that consumes the exception cannot be inferred by the absence of the exception.

However, in *Windows XP* and later, any registered Vectored Exception Handler will run before the Structured Exception Handler that *Windows* registers. This might be considered a bug in *Windows*. In this case the presence of a debugger that consumes the exception can be inferred by its absence.

2.5 DbgPrint

Similarly, despite the fact that the ntdll DbgPrint() function raises the DBG_PRINTEXCEPTION_C (0x40010006) exception, a registered Structured Exception Handler will not see it. Once again, the reason is that *Windows* registers its own Structured Exception Handler internally, which consumes the exception if a debugger does not do so. As such, the presence of a debugger that consumes the exception cannot be inferred by the absence of it.

However, as discussed previously, in *Windows XP* and later, any registered Vectored Exception Handler will run before the Structured Exception Handler that *Windows* registers and the presence of a debugger that consumes the exception can now be inferred by the absence of the exception. Further, a different exception is delivered to the Vectored Exception Handler if a debugger is present but has not consumed the exception, or if a debugger is not present. If a debugger is present but has not consumed the exception, then *Windows* will deliver the DBG_PRINTEXCEPTION_C (0x40010006) exception. If a debugger is not present, then *Windows* will deliver the EXCEPTION_ACCESS_VIOLATION (0xC0000005) exception. The presence of a debugger can now be inferred by either the absence of the exception, or by the value of the exception.

2.6 LoadLibrary

The kernel32 LoadLibrary() function is an unexpected method for debugger detection, but a simple and effective one. When a file is loaded in the presence of a debugger using the kernel32 LoadLibrary() function, and then freed, a handle remains open for that file. As a result, the file can no longer be opened for exclusive access. This fact can be used to infer the presence of the debugger.

Example code looks like this:

```
     mov   esi, offset l1
     push esi
     call LoadLibraryA
     push eax
     call FreeLibrary
     xor   ebx, ebx
     push ebx
     push ebx
     push 3
     push ebx
     push ebx
     push 80000000h
     push esi
     call CreateFileA
     inc   eax
     je    being_debugged
     ...
l1:  db    "myfile", 0
```

A less obvious method of achieving the same thing is to use the resource-updating APIs, specifically the kernel32 EndUpdateResource() function. The reason this works is because it eventually calls the kernel32 CreateFile() function to write the new resource table.

Example code looks like this:

```
     mov   esi, offset l1
     push esi
     call LoadLibraryA
     push eax
     call FreeLibrary
     push 0
     push esi
     call BeginUpdateResourceA
     push 0
     push eax
     call EndUpdateResourceA
     test eax, eax
     je    being_debugged
     ...
l1:  db "myfile", 0
```

2.7 NtQueryInformationProcess

As with the ProcessDebugPort class mentioned in [1], two other classes are similarly affected by arbitrary patching without checking the process handle: ProcessDebugObjectHandle and ProcessDebugFlags.

Example code for the ProcessDebugObjectHandle class looks like this:

```
     xor   ebx, ebx
     mov   ebp, offset l1
     push ebp
     call GetStartupInfoA
     ;sizeof(PROCESS_INFORMATION)
     sub   esp, 10h
     push esp
     push ebp
     push ebx
     push ebx
     push 1 ;DEBUG_PROCESS
     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
     ;ProcessDebugObjectHandle
     push 1eh
     push eax
     call NtQueryInformationProcess
     pop   eax
     test eax, eax
     je    being_faked
     ...
     ;sizeof(STARTUPINFO)
l1:  db    44h dup (?)
l2:  db    "myfile", 0
```

Example code for the ProcessDebugFlags class looks like this:

```
     xor   ebx, ebx
     mov   ebp, offset l1
     push ebp
     call GetStartupInfoA
     ;sizeof(PROCESS_INFORMATION)
     sub   esp, 10h
     push esp
     push ebp
     push ebx
     push ebx
     push 1 ;DEBUG_PROCESS
     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 1fh ;ProcessDebugFlags
```

```
      push eax
      call NtQueryInformationProcess
      pop  eax
      test eax, eax
      jne  being_faked
      ...
      ;sizeof(STARTUPINFO)
 l1: db   44h dup (?)
 l2: db   "myfile", 0
```

3. Hardware tricks

3.1 Execution timing

When a debugger is used to single-step through code, there is a significant delay between the execution of the individual instructions when compared to native execution. This delay can be measured using one of several possible time sources. These sources include the kernel32 QueryPerformanceCounter(), kernel32 GetSystemTime() and kernel32 GetLocalTime() functions, the winmm timeGetSystemTime() function, and interrupt 0x2A (also known as the KiGetTickCount() function).

4. Process Tricks

4.1 No import table

Windows NT and *Windows 2000* assume that an executable file contains an import table, and that as a result, kernel32.dll is loaded. Kernel32.dll can be loaded by importing a function directly from kernel32.dll, but it is also acceptable to import a function from another DLL that also imports from kernel32.dll (user32.dll, gdi32.dll, etc.).

Normally, if kernel32.dll is not present, a fault will occur at the location at which the context EIP points, because no page is mapped there. However, it is possible to change the value in the PE->ImageBase field to place the executable file in that location. Then, whenever the file is executed, it will receive control instead of causing a fault. Further, since ntdll.dll is always loaded, it is possible to make use of some of its functions, such as ntdll LdrLoadDll() and ntdll LdrGetProcedureAddress(), to resolve the required functions and execute normally.

4.2 Anti-debugging DLLs

Dynamically loaded DLLs are called initially with the DLL_PROCESS_ATTACH parameter. If they refuse to load, they will be called immediately again, but with the DLL_PROCESS_DETACH parameter. Statically loaded DLLs are also called with the DLL_PROCESS_ATTACH parameter. However, if they refuse to load, then the ntdll NtRaiseHardError() function will be called in order to display the message: 'The application failed to initialize properly'. Following that, the ntdll RtlRaiseStatus() function will be called.

In the absence of a debugger, this function will trigger an exception that cannot normally be intercepted, because all registered Structured Exception Handlers will have been removed already. However, if the topmost Structured Exception Handler is replaced, then it will be called via the ntdll RtlRaiseStatus() function call. This can allow a DLL to continue execution after a message that suggests that it terminated.

Example code looks like this:

```
      push esi
      xor  esi, esi
      fs:lodsd
      inc  eax
 l1: dec  eax
      xchg eax, esi
      lodsd
      inc  eax
      jnz  l1
      mov  d [esi], offset l2
      pop  esi
      ret
 l2: ...
```

In this case, l2 will gain control after the message box is dismissed.

4.3 TLS Callback

Thread Local Storage (TLS) callback is an old technique that remains relatively under-investigated. The following are some new extensions:

- The TLS callback array can be altered (later entries can be modified) and/or extended (new entries can be appended) at runtime. Newly added or modified callbacks will be called using the new addresses. There is no limit to the number of callbacks that can be placed. This technique has been disclosed publicly [2].

  Example callback code looks like this:

  ```
  l1: mov d [offset cbEnd],offset l2
      retn
  l2: ...
  ```

  The callback at l2 will be called when the callback at l1 returns.

- TLS callback addresses can point outside of the image – for example, to newly loaded DLLs.

  Example callback code looks like this:

  ```
  l1: push offset l2
      call LoadLibraryA
      mov      [offset cbEnd], eax
      ret
  l2: db       "tls2", 0
  ```

In this case, the 'MZ' header of tls2.dll will be executed when the callback at l1 returns. The file header can be made executable despite DEP, using the SectionAlignment trick described in [3]. This allows the code to run without error.

- TLS callback addresses can contain RVAs of imported addresses from other DLLs if the import address table is altered to point into the callback array. Imports are resolved before callbacks are called, so imported functions will be called normally when the callback array entry is reached.
- TLS callbacks receive three stack parameters, which can be passed directly to APIs. The first parameter is the ImageBase of the host process. It could be used by APIs such as the kernel32 LoadLibrary() or kernel32 WinExec() functions. The ImageBase parameter will be interpreted by the kernel32 LoadLibrary() or kernel32 WinExec() functions as a pointer to the filename to load or execute. By creating a file called 'MZ[some string]', where 'some string' matches the host file header contents, the TLS callback will access the file without any explicit reference. Of course, the 'MZ' portion of the string can also be replaced manually at runtime, but many APIs rely on this signature, so the results of such a change are unpredictable.
- TLS callbacks are called whenever a thread is created or destroyed (unless the process calls the kernel32 DisableThreadLibraryCalls() or the ntdll LdrDisableThreadCalloutsForDll() functions). This includes the thread that is created by *Windows* when a debugger attaches to a process. The debugger thread is special in that its entrypoint does not point inside the image. Instead, it points inside kernel32.dll. Thus, a simple debugger detection method is to use a TLS callback to query the start address of each thread that is created.

  Example callback code looks like this:

```
   push eax
   mov  eax, esp
   push 0
   push 4
   push eax
   ;ThreadQuerySetWin32StartAddress
   push 9
   push -2 ;GetCurrentThread()
   call NtQueryInformationThread
   pop  eax
   cmp  eax, offset l1
   jnb  being_debugged
   ...
l1: <code end>
```

- Since TLS callbacks run before a debugger can gain control, the callback can make other changes, such as removing the breakpoint that is typically placed at the host entrypoint. When combined with the ntdll DbgBreakPoint() function patch, the result is a file that cannot be debugged by ordinary means. The debugger will attach to the debuggee, and then wait for the exception which will never occur. Using Ctrl-C to break in will work well enough to look at the code, but breakpoints that are placed within the other threads will not activate.

  Example callback code looks like this:

```
   push offset l2
   call GetModuleHandleA
   push offset l3
   push eax
   call GetProcAddress
   push eax
   push esp
   push 40h  ;PAGE_EXECUTE_READWRITE
   push 1
   push eax
   xchg ebx, eax
   call VirtualProtect
   mov  b [ebx], 0c3h
   ;<val> is byte at l1
   mov  b [offset l1], <val>
   pop  eax
   ret
l1: <host entrypoint>
   ...
l2: db  "ntdll", 0
l3: db  "DbgBreakPoint", 0
```

  Currently, it seems that no debugger handles this case. However, the fix is very simple, and increasingly necessary. It is a matter of inserting the breakpoint on the first byte of the first TLS callback instead of the host entrypoint. This will allow an exception to be raised as usual. However, care must be taken regarding the callback address, since as noted above, the address may be the RVA of an imported function. Thus, the address cannot be taken from the file header. It must be read from the image memory.

In part three of this article next month we will look at some miscellaneous anti-debugging tricks, as well as a range of tricks that target specific debuggers.

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

REFERENCES

[1] Ferrie, P. Anti-unpacker tricks – part one. Virus Bulletin, December 2008, p.4.

[2] Self-modifying TLS callbacks. http://www.openrce.org/blog/view/1114/Self-modifying_TLS_callbacks.

[3] Ferrie, P. Anti-unpacker tricks. 2008. http://pferrie.tripod.com/papers/unpackers.pdf.

FEATURE 1

A DAY IN THE LIFE OF AN AVERAGE USER

Gabor Szappanos
VirusBuster, Hungary

The idea for this article came to me as I was reading a tabloid newspaper to pass the time while travelling. The paper quoted an AV company which estimated that an average computer user is flooded with new malware threats every second, and that they are attacked by malware tens of thousands of times every day. Now, I thought that I had a more-or-less clear picture of what malware is spreading out there [1], and based on my experience, I felt that the estimates made in the newspaper were too high. To find out who was right, I decided to do some research into the dangers of being an average user.

USER PROFILE

For the sake of this experiment, we picked a Hungarian computer user (left). Any similarity to any persons, living or dead, is purely coincidental.

The activities of our user cover general email traffic and Internet browsing, with Internet connection at work and a broadband connection at home. Our user has not registered on any pornographic websites and does not use Viagra and the like (this is not a boast, just a statement of fact that is relevant to the user's profile, and which will have consequences for the level of threat to which the user is exposed). As a result, our subject is less vulnerable than some users, and we must take into account the fact that some of the threats measured here will potentially be underestimated.

THE RISKS OF BEING CONNECTED

At home our subject is connected to the Internet. Without even sitting in front of his keyboard he is a target for external attacks simply because he is connected. These attacks come from lurking network worms. Our subject is not stupid enough to leave his home PC wide open to attack and has installed the latest security updates and a firewall which protect him from most of these lurking threats. However, we should take these threats into consideration because we know that there are many users who do not follow the same security practices. Simple worm traps located on the user's ISP were used to measure the number of attacks from network worms. The number of threats captured by an SMB trap [2] on the day in question is shown in Table 1.

Threat	Prevalence
Worm.Opaserv.AV	3
Worm.Opaserv.AK	3
Win32.Heretic.1986 + Opaserv.AK	2
IRC.Flood.AO	2
Worm.Opaserv.AB	2
IRC.Flood.AS	2
BAT.Flood.BS	1
Worm.Opaserv.AS	1
Hacktool.IpcScan.C	1
Backdoor.ServU-based.B	1
Tool.PsExec.A	1
Hacktool.SQLScan.C	1
Worm.Opaserv.BC	1
IRC.BNC.N	1
RiskTool.HideWindows.AB	1
Worm.Opaserv.AX	1
Win32.Parite.B	1
I-Worm.Opaserv.J	1
Backdoor.ServU-based.H	1

Table 1: Number of threats captured in SMB trap.

Altogether 34 attacks were recorded from 27 different IP addresses. The list is dominated by ancient Opaserv variants. One of Opaserv's attack vectors is an old vulnerability (MS00-072) which has long since been fixed, but it also attacks weak admin passwords. I would love to be able to say that this is no longer a threat because every computer has already been patched and no one is stupid enough to set a weak admin password like 'admin' or '123'. However, reliable sources suggest that 97% of the population does not reside at the genius end of the scale [3], and with weak passwords a common occurrence, these Opaserv variants remain a threat.

NEPENTHES TRAP

More recent network worms, predominantly bots, use a newer (and ever-increasing) set of *Windows* vulnerabilities to infect network-connected computers. These vulnerabilities are simulated (and the attacks are captured) by honeypots like *nepenthes* or *mwcollect*. This is a different vulnerability window for an average user and it is fair to enumerate the threats coming from it separately. The top 15 attacks observed on the day in question are shown in Table 2.

Altogether there were 225 successful attacks[1] from 98 different IP addresses. Most of the captures were some

[1] By 'successful attack' we mean that the goat PC was attacked, and the connect-back code downloaded the attacking worm successfully.

Threat	Prevalence
Win32.Virut.Gen	31
Worm.Kolabc.DW	30
Trojan.DR.Agent.EXVG	21
Worm.SdBot.ACIV	16
Worm.Agobot.WPTY	16
Worm.Kolabc.DO	10
Worm.Rbot.ACWL	9
Worm.Agobot.WPUU	9
Worm.Rbot.MCH	8
Worm.Rbot.MCG	7
Worm.Akbot.CE	7
Worm.Allaple.Gen	7
Worm.Agobot.WPUM	7
Worm.Allaple.AA	5
Backdoor.Allaple.Gen.2	5

Table 2: Newer network worms.

flavour of bot, and even the Virut variants were infections on top of Rbot or Allaple variants.

On rough average, a user is subjected to a successful attack from such threats once every six minutes.

Unlike the Opaserv variants captured by the *Samba* traps, which would not reach any user with the relevant security updates applied, these worms are direct threats to our user. Deploying the latest security patches and using a firewall can decrease the user's vulnerability, but a friendly ISP which filters out the *Windows* ports used by these worms can decrease the threat by several orders of magnitude. As a comparison, on a different ISP (with filtering in place) we recorded about four attacks per day using the same traps and set ups – all of them coming from an emerging new threat called SQLSlammer.

EMAIL ATTACKS

Email is still one of today's major attack vectors – it no longer serves primarily as a medium for self-spreading worms, but for seeding new trojan versions. However, there are still a couple of old friends in the playground.

Sitting behind corporate protection in the workplace and having an ISP that provides email filtering at home protects our subject from most of these email attacks. As this configuration is typical for most users, it would not be appropriate to include these threats in our calculations. However, for interest's sake, Table 3 shows a list of the top threats blocked by the email filter of a large Hungarian ISP (which happens to be our user's ISP).

The list also shows the discovery date of these worms (which is also almost exactly the date on which protection

Threat		Prev.	Discovery
[1]	I-Worm.Zafi.B	2,694	(2004.06)
[2]	I-Worm.Netsky.Q1	2,002	(2004.03)
[3]	Exploit.IFrame.B	1,744	
[4]	I-Worm.Zafi.D	1,005	(2004.12)
[5]	Win32.Virut.Gen.4	980	
[6]	I-Worm.Netsky.Q2	822	(2004.03)
[7]	I-Worm.Netsky.R	493	(2004.03)
[8]	Trojan.FakeAlert.Gen!Pac	423	
[9]	I-Worm.Bagle.LC	299	(2006.12)
[10]	I-Worm.Netsky.D3	280	(2004.03)
[11]	I-Worm.Bagle.ZIP.Gen.3	234	(2006.06)
[12]	Worm.P2P.VB.CIL!CME-24	198	(2006.01)

Table 3: Top threats blocked by the email filter of a large Hungarian ISP.

was released for them). Only two of the top ten are less than two years old, and the majority of the list comprises worms that are at least four years old. However, as discussed, this category of threats is unlikely to reach an average user.

Other email-borne attacks are filtered out by spam filters. A spam filter can serve as a first line of defence against email-based attacks [4].

SPAM

While spam mostly transmits messages that keep the global underground economy rolling, it is also a part-time distributor of malware.

Figure 1 shows the distribution of spam types received by our subject. He has not taken any active steps to increase his chances of receiving spam (e.g. he is not using his email address as a spam trap). Users with a more active online social life may experience larger volumes of spam. For the sake of better statistics, the numbers were measured

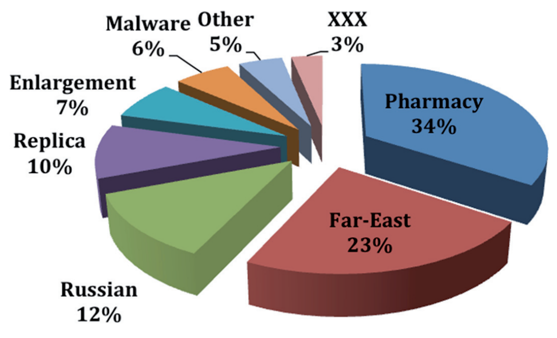


Figure 1: Types of spam messages received.

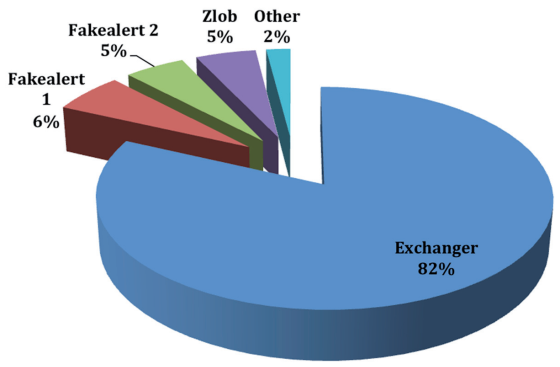
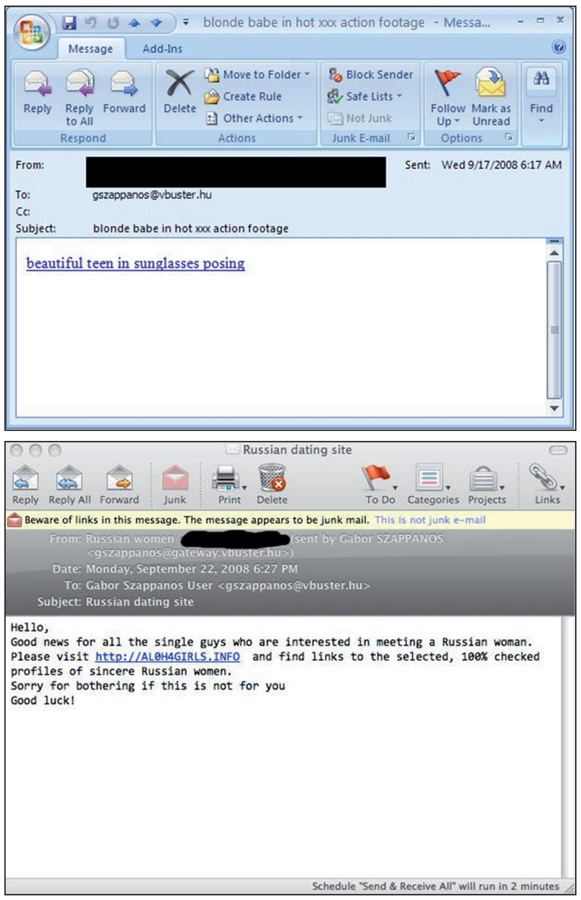


Figure 2: Distribution of malware within spam.

over a one-week period and averaged (a total of 1,740 messages, 250 daily). In total, 6% of the spam messages received by our user were associated with malware propagation.

The distribution of malware within these messages is shown in Figure 2. These include both malicious attachments and spammed links pointing to (MPacked) malware distribution sites. Figure 3 shows an example of how difficult it can be to distinguish between a link that points to malware and one that points to a 'conservative' adult site. Altogether 105 malware messages were received within a week (15 per day), most of which were Exchanger, the Fakealert and Zlob.



From: [redacted] Sent: Wed 9/17/2008 6:17 AM
To: gszappanos@vbuster.hu
Cc:
Subject: blonde babe in hot xxx action footage

beautiful teen in sunglasses posing

Russian dating site

Beware of links in this message. The message appears to be junk mail. This is not junk e-mail

From: Russian women [redacted] (sent by Gabor SZAPPANOS <gszappanos@gateway.vbuster.hu>)
Date: Monday, September 22, 2008 6:27 PM
To: Gabor Szappanos User <gszappanos@vbuster.hu>
Subject: Russian dating site

```
Hello,
Good news for all the single guys who are interested in meeting a Russian woman.
Please visit http://AL0H4GIRLS.INFO  and find links to the selected, 100% checked
profiles of sincere Russian women.
Sorry for bothering if this is not for you
Good luck!
```

Schedule "Send & Receive All" will run in 2 minutes

Figure 3: Which one of these points to malware? (the message captured on OSX is safe – at least from malware).

DOWNLOAD UPDATES

In this investigation we are looking at a day in the life of our subject, but it is certainly not the first day. He has a history, and he may have been infected before. If there is already malware installed on his PC it will keep updating itself.

We could not simulate entirely the level of infection of our subject, but we got some help from our virus lab, where the update URLs of common malware families are monitored daily. Table 4 shows the most frequently updated malware domains on the day in question.

Most of these are related to online games, which are not very popular in Hungary, so while these may be a more significant threat to the global population, they are not relevant to our target individual.

URL	Prev.	Family	Location
try-count.net	151	Tibs	Moscow, Russia
sum4count.net	33	Tibs	Moscow, Russia
user1.16-m8.net	32	DL.Small	Zhanjiang, China
ee1.tu-sg.info	24	OnlineGames	Yiwu, China
webair.com	7	Zlob	Jericho, NY, USA
7894234.cn	7	OnlineGames	Shanghai, China
sql.78-11.net	6	OnlineGames	Beijing, China
23488ss.cn	5	OnlineGames	Shanghai, China
zango.com	4	Adware.Zango	Bellevue, WA, USA
111.hfdy2525.net	4	OnlineGames	Ruian, China
www.fsjinqu.cn	4	QQPass	Beijing, China

Table 4: Most frequently updated malware domains.

Other risk factors we have not measured (because our subject is not involved in them) include:

- IRC: IRC is used mostly by bots for communication purposes, and less frequently for seeding.
- Instant messaging: some common families mass-distribute download links on IM networks.
- P2P file sharing: our target kept a loose eye on P2P file-sharing networks during the research week, but since he is not involved in the use of cracked software, there is no measurement data on this threat.
- Drive-by exploits: these are an increasingly important vector for malware distribution. A large number of legitimate websites have been hacked to include downloader scripts (pointing mostly to MPack distributions). This threat was excluded because it was not possible to measure its prevalence accurately – but during the last week at least three popular legitimate Hungarian sites were found to be distributing malware via this method. So this is a real threat, but not measured here.

Although not measured in this investigation, all of these risk factors raise the overall threat level of an average user (if he happens to be using these technologies).

CONCLUSION

Are we being attacked every minute? Despite the fact that most of the threats have been underestimated, this investigation has shown that we are far from being threatened by thousands of trojans every day.

The investigation has shown that we face many threats every day coming at us from different directions – but with reasonable care these risks can be minimized.

REFERENCES

[1] Szappanos, G. What is out there? Virus Bulletin, January 2006, p.10. http://www.virusbtn.com/pdf/magazine/2006/200601.pdf.

[2] Overton, M. Worm charming: taking SMB Lure to the next level. Proceedings of the Virus Bulletin International Conference, 2003.

[3] Bontchev, V. Anatomy of a Virus Epidemic. Proceedings of the Virus Bulletin International Conference, 2001.

[4] Overton, M. Canning more than Spam with Bayesian filtering. Proceedings of the Virus Bulletin International Conference, 2004.

CALL FOR PAPERS

CALLING ALL SPEAKERS: VB2009 GENEVA

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.

A list of topics suggested by the attendees of VB2008 can be found at http://www.virusbtn.com/conference/vb2009/call/. However, please note that this list is not exhaustive, and the selection committee will consider papers on these and any other anti-malware and anti-spam related subjects.

SUBMITTING A PROPOSAL

The deadline for submission of proposals is **Friday 6 March 2009**. Abstracts should be submitted via our online abstract submission system. You will need to include:

- An abstract of approximately 200 words outlining the proposed paper.
- Full contact details with each submission.
- An indication of whether the paper is intended for the technical or corporate stream.

The abstract submission form can be found at http://www.virusbtn.com/conference/abstracts/.

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

Authors are advised that, should their paper be selected for the conference programme, the deadline for submission of the completed papers will be Monday 8 June 2009, and that they must be available to present their papers in Geneva between 23 and 25 September 2009.

Any queries relating to the call for papers should be addressed to editor@virusbtn.com.

FEATURE 2

ADVANCING MALWARE TECHNIQUES 2008

Alisa Shevchenko
Independent researcher, Russia

The following article is an overview of protection bypassing techniques found in current *Windows*-targeting malware. It is a sequel to a previous paper on the topic [1] and presents a basic, but far from exhaustive, overview of current malware survival tricks.

RESEARCH DETAILS

Research has been conducted mainly upon a random list of malware found in the wild over the last six months. The list includes malicious proxy samples (such as Trojan-Proxy.Win32.Agent.xd), simple downloading malware (such as Trojan-Downloader.Win32.Small.htz), and anti-virus-killing trojans (specifically, various modifications of Trojan.Win32.KillAV). In addition, a number of tricks were found inside commercial malware, such as the so-called 'Emotions Loader', a malicious downloader bot heavily traded on the Russian black market and widely distributed in the wild.

Since tracking malware is not in my daily routine, I was not in a position to distinguish brand new 'zero-day' techniques from older ones in my research; however, I am sure of the following:

- The techniques listed in this review are relatively new (they have appeared and/or have been popularized during the last year).
- The techniques work very well (since malware writers are still implementing them).
- The techniques are widespread (most of them are found in mass-distributed malware rather than in targeted malware).

The techniques are listed below with brief descriptions and are grouped by their purpose.

ANTI-MALWARE PROTECTION BYPASSING

These are techniques that are used to bypass various security software and technologies, including anti-virus behavioural heuristics, anti-virus products in general, Host Intrusion Prevention Systems (HIPS) and personal firewalls.

Kernel function hook

A common approach to bypassing a personal firewall (to achieve a silent malicious download, for example) consists of implementing some kind of brute force technique, from attempting to kill a firewall process or remove its required API hooks, to injecting into firewall-trusted modules.

A technique I encountered recently is curious in that the only modification made to the state of the operating system is a single kernel function hook. With such an approach, a firewall will never know it is being fooled, and a HIPS will not detect a process injection attempt.

The following is a summary of the technique in intuitive pseudo code:

```
If NdisRegisterProtocol address lies outside of
ndis.sys //which means it's hooked
{
   Hook IoGetCurrentProcess:
       If the return value of the original
IoGetCurrentProcess corresponds to the bypassing
malware's own process
       and
       If IoGetCurrentProcess stack return address
falls around NdisRegisterProtocol
//which means the caller is NdisRegisterProtocol hook
master = most probably a firewall!
         Return fake process pointer //a trusted one.
}
```

The realization of this technique is quite elegant, exploiting the fact that many personal firewalls hook NdisRegisterProtocol to guarantee the securing of newly added protocols. While feeding a firewall a trusted process pointer in place of a malicious process pointer is an obvious idea, the idea of locating a firewall module via its own hook is cute, resulting in an unobtrusive, compact and quite generic approach to firewall bypassing.

Sending IO control code

Another technique, which also demonstrates the unobtrusive trend of protection bypassing, consists of forcing a security application to quit, or otherwise controlling it, by means of sending its own driver a valid IO control code. The necessary IO control codes may be retrieved by means of reverse engineering.

The concern in both these cases is that malware writers have arrived at the idea of 'turning the enemy's armour into a weakness' – in other words, defeating a security measure by its own specifics, instead of fighting against it. Why bother to fight, if you can make the enemy trip itself up?

In respect of the trend, security software vendors are urged to consider their products' architecture, including component communication mechanisms, and to protect their binary code.

Process termination from kernel mode

Some malware writers still prefer to kill the security program completely rather than bypass it. But, given that most security vendors now equip their products with some kind of self-protection mechanism, killing a protected process can be a hard task, even from kernel mode.

The following technique is an advanced process termination from kernel mode, which allows basic self-protection mechanisms to be bypassed. It consists of initializing (via KeInsertQueueApc) an asynchronous procedure call to ZwTerminateProcess for the process to be killed. In this case, the process termination API is called in the context of the process being killed and not from a third-party process which may be restricted by self-protection code.

Note: some anti-virus vendors ignore the challenge of checking/securing kernel events, justifying this by the fact that once a piece of malware gets into the kernel, fighting it is pointless. While the statement is reasonable, the conclusion is arguable. Even though all known methods of getting into the kernel are monitored by most anti-malware products in the first place, new methods continue to appear. Malware often manages to get into the kernel despite our best attempts to prevent it. Thus, considering kernel events (checking them by code heuristics or against behavioural patterns etc.) is no less important than securing the kernel entry gates or monitoring basic system events.

SURVIVING SYSTEM RESTART

Image File Execution Options registry key

Current malware in the wild makes extensive use of the 'Image File Execution Options' registry key. The key, located under HKLM\\Software\\Microsoft\\Windows NT\\CurrentVersion\\, is normally in charge of keeping persistent execution options for various standalone executables. Among them is the option to always run a certain executable under a debugger, the latter defined explicitly in the form of a path to an unvalidated .exe under the 'Debugger' value in the executable subkey.

One of the ways to misuse this key is for a piece of malware to insert a path to itself into the 'Debugger' property for a common-use executable, resulting in the malware being executed each time the executable is run.

Modifying service registry key

This technique is pretty old and simple, but still widely used by malware in the wild, which may mean that some security vendors fail to flag it by heuristic or behavioural signature.

The technique consists of modifying an existing service registry key to provide a malicious component startup, putting the path to a piece of malware into the 'ImagePath' value instead of a valid service executable. An example of a service which runs by default and is very rarely used by an end-user (thus making it a perfect spoofing goal) is the Scheduler service, located under the SYSTEM\\CurrentControlSet\\Services\\Schedule key.

MINOR TRICKS

Disabling WSFP

An 'old, new' approach to bypassing Windows System Files Protection is simply to disable it temporarily, by calling an undocumented API function named SetSfcFileException (ordinal 5) from the sfc_os.dll. Making use of this API provides a way to replace system files, patch them or upgrade them with exportable malicious functionality.

Previously malware writers patched the sfc_os.dll to bypass file protection.

Calling alternative API functions

Some malware still calls alternative API functions from ntdll.dll instead of the common-use APIs in order to fool anti-virus heuristics. As an example, a piece of malware may call NtDuplicateObject instead of DuplicateHandle to obtain the necessary access rights to a process via duplicating its handle, and subsequently kill a process or inject into it. Another example is to call LdrLoadDll instead of LoadLibrary to execute a malware component.

RECOMMENDATIONS

The following are some recommendations for security software vendors and researchers to help keep ahead of the protection bypassing game:

- Take time to thoroughly analyse regular downloading malware and their propagation vectors. Since downloading malware represents one of the basic tiers of the computer criminal industry, their creators are highly motivated to provide effective and up-to-date mechanisms for bypassing security software.
- Monitor and analyse publicly traded commercial malware. For clear reasons it is the most quickly developing category of malware, yielding only to privately traded and targeted malware in respect of technological advance.
- Track researcher blogs and forums. Researchers from Russia and China seem to be particularly interested in advanced malware technologies and security software bypassing.

REFERENCE

[1] Shevchenko, A. Bypassing and enhancing live behavioral protection. InSecure Magazine #17.

The author welcomes comments/suggestions at contact@alisa.sh.

PRODUCT REVIEW

MICROWORLD ESCAN INTERNET SECURITY SUITE 10

John Hawes

MicroWorld Technologies was founded in 1994, and is incorporated in New Jersey with its development team based in India, and offices and partner companies worldwide. The firm specializes in security, providing a wide range of solutions and services on multiple platforms. Its best-known and flagship product range is the *eScan* range of security software.

eScan has been a regular entrant and pretty consistent performer in VB100 testing since 2003. In the course of five years of testing, the product has picked up 16 VB100 awards, with only a handful of false positives upsetting things on a few occasions in the last few years. Over this time, the product has evolved from providing an alternative front end to multi-engine scanning to the current complete Internet suite, with the *Kaspersky* detection engine at the core of the anti-malware protection. A slick new look accompanies the upgraded set of functions and protection features, and after putting it through its paces in the last VB100 comparative, we were keen to take at deeper look at what it had to offer.

WEB PRESENCE, INFORMATION AND SUPPORT

The main online home of *MicroWorld* is at www.mwti.net, a simple, unflashy site with the tagline 'We add confidence to computing'. The sober, pared-down nature of the home page is offset by a glossy, animated advertisement for the product under review this month, *eScan 10*, which is due for release very soon. The remainder of the page is simply rendered and information-packed. The left pane carries a comprehensive menu of the site's offerings, with the product range highlighted and categorized by user type, product type and operating system. Also here are links to support, product purchasing, downloads, licensing and updates, as well as some company information. The central panel runs through the product range again, with links leading to detailed pages on each, and below this is a cluster of company news items and press releases. The right-hand pane is more support-oriented, with a 24/7 toll-free US support number prominent, closely followed by a link to support forums. Below this, information is provided on the latest threats and product updates, and there is the option to sign up to a newsletter for those wanting to keep up to date with this kind of news.

The product section is very clear and thorough, with each market sector (home-user, small business and enterprise) treated to a thorough list of the available solutions with full details on the protection they provide. The range itself is pretty extensive, with support for a variety of *Windows* requirements – from the desktop suite under review here through enterprise versions with full management and reporting systems, to mail and web gateway solutions – while *Linux* users are similarly well catered for, the product list again running the gamut from the desktop to the gateway.

The download section sits behind a form requesting some personal information, such as your email address and which product you are interested in downloading, but this seemed not to work for us.

The support section suggests users contact the firm via email if possible, but also provides a chat system, with links to *MSN Messenger* or *Yahoo! Chat* for those that need them, as well as free telephone support for all users – an admirable provision for users in dire need. For those with less urgent issues, the forum section provides a reasonable collection of FAQs alongside the busy tech support area, where answers seem to be provided promptly and graciously.

More in-depth product information seemed a little hard to come by – with no search option obvious on the website, a rummage around eventually turned up manuals for the products, the links tucked away at the bottom of the details pages. The manuals seemed fairly thorough and clearly written, if a little short on colour and illustration, and organized more by function than task. Help within the product itself, we later found, had no local data but instead connected to the web, calling up the appropriate page of a wiki-based system – an interesting new direction for such functionality. Sadly, as the product under test is still in the final stages of beta, no manual was available and much of the wiki remained unpopulated, so no assessment could be made of its quality and usefulness.

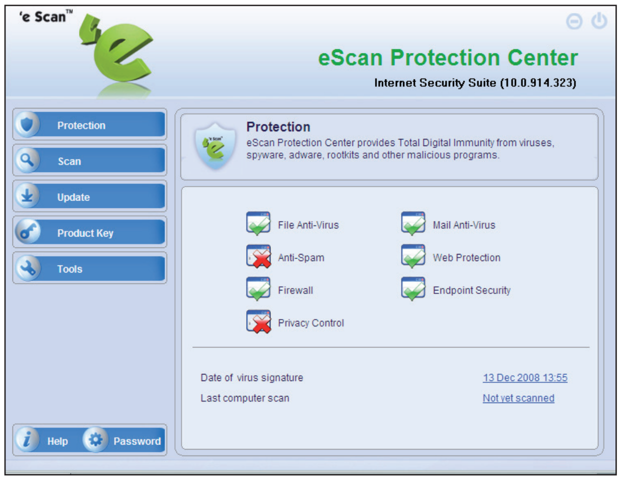
Returning to the main website, the virus information area presents a cursory malware encyclopaedia, not over-stocked with entries, but lucid and detailed on those covered. The company sections provide a selection of titbits on the company's past and achievements, including its VB100 awards, as well as lists of events being attended by company representatives in the upcoming months, job vacancies, and more complete contact details for the various branches.

One of the most useful things to be found here is a free version of the on-demand scanner, *MWAV*. This neat little tool provides the full functionality of the on-demand part of the product, including cleaning, quarantining and updates, without the need for a full installation process. It can be run in conjunction with installed anti-virus without much further effort, providing a useful alternative to online scans and other such extra layers of security reassurance. Most of the company's other products also seem to be offered on a free trial basis for interested users.

INSTALLATION AND CONFIGURATION

Moving off the web and into the lab, we took a second look at the new *ISS* product, having given it a quick run-through in the recent VB100 comparative. The set up process is pretty standard, going through the usual options and then taking about a minute or so to perform its copying and configuration processes before launching an immediate scan of vital areas, the system folders and registry. A reboot was then required to complete the installation, after which a popup declared *eScan* had provided 'The world's first real-time email and webscanner', while a swift and unobtrusive update of definitions etc. zipped along in the background. Beyond that, no further manual configuration seemed to be required – all firewall rules etc. are applied automatically at a default level with none of the customary requests for detailed technical information seen in many products these days, making *eScan* suitable for the most technophobic of users. With everything set up and safe, we fired up the main interface to explore what was on offer for those with more particular and exacting requirements.

The main interface is pleasantly straightforward, with a simple list of sections down one side and checkboxes in the main window to indicate the status of various modules. Some of the main options seem a little less vital than others: while the 'Protection' tab covers a seriously wide range of controls, 'Update' is perhaps less of a major issue, at it should mostly be automated, while one wouldn't expect to use the 'Product key' section more than once a year – perhaps it disappears once a full licence is applied. Looking into the update tab, a huge range of configuration options are available beyond the usual



eScan Protection Center

Internet Security Suite (10.0.914.323)

Protection

eScan Protection Center provides Total Digital Immunity from viruses, spyware, adware, rootkits and other malicious programs.

✓ File Anti-Virus
✓ Mail Anti-Virus
✗ Anti-Spam
✓ Web Protection
✓ Firewall
✓ Endpoint Security
✗ Privacy Control

Date of virus signature 13 Dec 2008 13:55

Last computer scan Not yet scanned

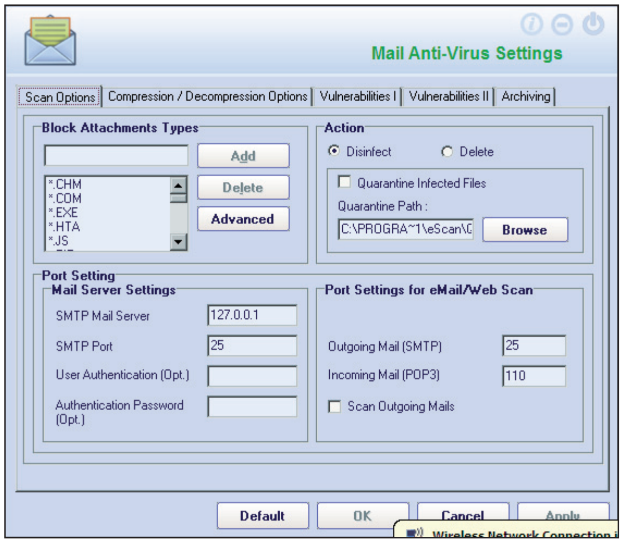
basics of automatic/manual and frequency options. Update sources and connection protocols can be set, including local network updates, schedules can be fine-tuned to great depth, and a range of post-update activities, including mail notifications and customs execution of commands, are also available, providing enterprise-grade configurability to those who want it. The licence key tab, however, simply offered a dialog box to enter a licence key.

A quick glance through the remainder of the main tabs showed a pleasant continuity of design, with initial pages clear and simple, providing basic data on the module in question and buttons to access detailed configuration and reporting options. We looked through some of these in more depth, starting with the system protection tab.

SYSTEM PROTECTION AND MALWARE DETECTION

The anti-malware components are controlled from the first two tabs, the first of which covers real-time monitoring and the second on-demand scanning. The on-demand tab provides a good list of default scan types, covering core areas such as memory, registry and system folders, the local file system, spyware and adware, USB drives and further custom options. A scheduling system allows simple set up of multiple types and depths of scan. Default settings tend towards the thorough, and the scanning speeds recorded in the recent VB100 test reflect this thoroughness, but there are plenty of options to exclude archives, areas, file types and so on, as well as prioritization options which can speed scans up or minimize system impact as required – such impacts were barely noticed, and on-access times in our earlier test showed pretty decent overheads. Even running on a fairly low-powered test notebook, no slow down in operation was evident to the naked eye.

Detection, mainly provided by the *Kaspersky* engine with some enhancements of *MicroWorld*'s own, is superb. The product, like many using the *Kaspersky* engine, has a pretty decent record in our VB100 testing. Undetected items are vanishingly rare, even in the more obscure test sets, and WildList misses almost unheard of, even when complex and difficult polymorphic viruses make their way onto the list. The recent addition of new, freshly updated sets of trojans into our test sets has shown up deficiencies in some products, but *eScan* has maintained an excellent standard, regularly catching more than 90 per cent of the samples with the standard scanner alone, putting it in the top handful of products in this regard. On the few recent occasions on which *eScan* has failed to achieve a VB100 award, it has been relatively minor false positive incidents that have let it down.



Mail Anti-Virus Settings

Scan Options | Compression / Decompression Options | Vulnerabilities I | Vulnerabilities II | Archiving

Block Attachments Types: *.CHM, *.COM, *.EXE, *.HTA, *.JS

Action: [✓ Disinfect] Delete

☐ Quarantine Infected Files

Quarantine Path : C:\PROGRA~1\eScan\Q

Port Setting

Mail Server Settings

SMTP Mail Server	127.0.0.1
SMTP Port	25
User Authentication (Opt.)	
Authentication Password (Opt.)	

Port Settings for eMail/Web Scan

Outgoing Mail (SMTP)	25
Incoming Mail (POP3)	110

☐ Scan Outgoing Mails

Of course, with the vast number of new malicious items being seen these days, simple signature-based detection is increasingly sidelined in favour of heuristics and other techniques. While the *Kaspersky* research lab has an excellent record of speed and throughput, keeping up with the flood at a remarkable pace, many products, including *Kaspersky*'s own suites, have begun to introduce more advanced behavioural monitoring and HIPS systems into their suites. Although no such options are evident in *eScan*, the few items we managed to find that were not detected by the standard scanners were blocked instantly on attempting to execute by some more sophisticated heuristic or emulation techniques – a simple message is presented, warning that the files in question are suspicious, and the user is given the option to allow them to run if trusted. Where the monitor has been protected with an administrator password, this password is required before such potentially unsavoury items are granted access to the system.

Anything not spotted by this protective layer would then come up against the firewall, which seems to operate well with sensible defaults and minimal interruption of the user with requests for permissions. Again, configurability is enormous, with the default 'limited filter' setting easily switched into a more interactive mode for those wishing to monitor events for themselves, and simple allow all/ block all buttons permitting easy lockdown or opening up of the system. The advanced tabs offer all the expected configurability, laid out and controlled with admirable clarity and simplicity.

The mail malware filtering is similarly configurable. The mail scanner allows attachment file types, multi-extension files, and even files with specified strings in the name to be blocked automatically, introducing an element of content filtering and policy enforcement to the mix. It can be set to decompress and scan compressed attachments if required, and has a number of anti-exploit techniques including the blocking of HTML mails containing scripts. All are readily customizable.

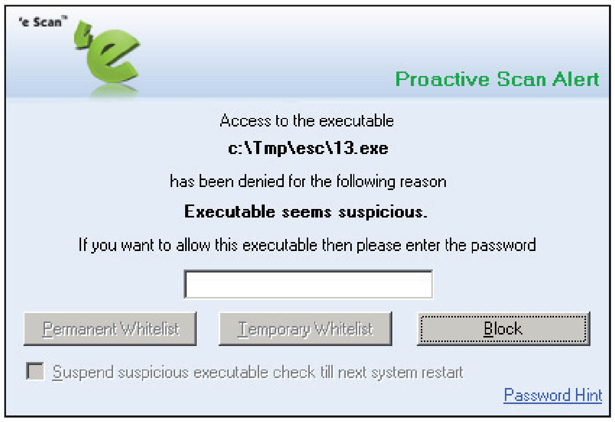
All these areas offer in-depth logging, with the firewall particularly well provided for in this area – detailed summaries and reports can be generated on recent network activity and what filtering was applied, with a graph generator included. Current activity can also be monitored in real time using a TCP viewer utility included with the suite. This is a nifty little tool which provides details of all processes and connections passing through the firewall, and allows the user to kill any undesired processes.

The remainder of the 'Protection' section, while providing some protection from malware, also addresses a range of other security issues, and will thus be analysed in the next section.

OTHER FUNCTIONALITY

Without wanting to play down the main anti-malware and anti-hacking components of this suite, which are clearly of high quality and solidly implemented, they are fairly standard and to be expected in a security product. Much of the beauty of this latest iteration of *eScan* lies in the many extras to be found in the additional tabs and in advanced configuration controls of the main modules.

The anti-spam component has become another basic and expected part of any Internet security suite, and it is not omitted here. The filter, when fired up, checks out installed mail software including *Outlook* and *Thunderbird*, with many other tools supported, and boasts 'Non Intrusive Learning Pattern' checking. The controls and configuration are extremely clear and easy to navigate, and the advanced options splendid – there is an option (enabled by default) to classify all mails containing Chinese or Korean characters as spam, which would reduce my personal spam level by



Proactive Scan Alert

Access to the executable

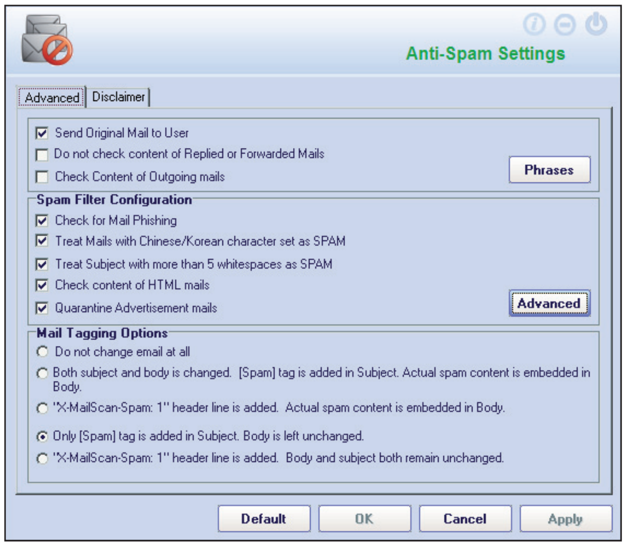
c:\Tmp\esc\13.exe

has been denied for the following reason

Executable seems suspicious.

If you want to allow this executable then please enter the password

☐ Suspend suspicious executable check till next system restart



Anti-Spam Settings

Advanced | Disclaimer

- [✓] Send Original Mail to User
- [ ] Do not check content of Replied or Forwarded Mails
- [ ] Check Content of Outgoing mails

Phrases

Spam Filter Configuration

- [✓] Check for Mail Phishing
- [✓] Treat Mails with Chinese/Korean character set as SPAM
- [✓] Treat Subject with more than 5 whitespaces as SPAM
- [✓] Check content of HTML mails
- [✓] Quarantine Advertisement mails

Advanced

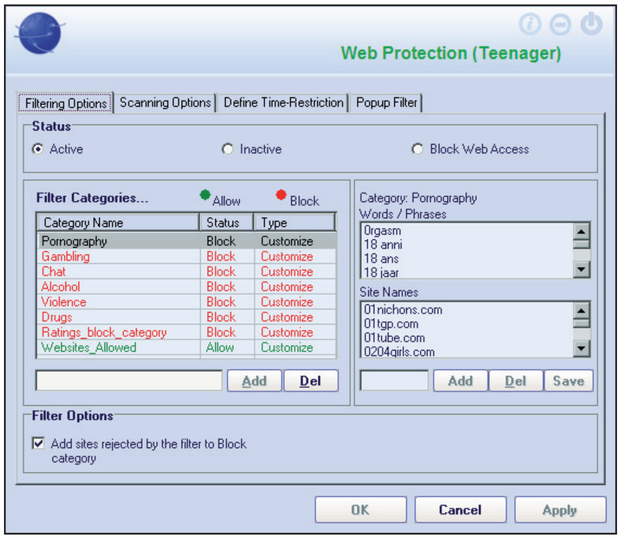
Mail Tagging Options

- ( ) Do not change email at all
- ( ) Both subject and body is changed. [Spam] tag is added in Subject. Actual spam content is embedded in Body.
- ( ) "X-MailScan-Spam: 1" header line is added. Actual spam content is embedded in Body.
- (✓) Only [Spam] tag is added in Subject. Body is left unchanged.
- ( ) "X-MailScan-Spam: 1" header line is added. Body and subject both remain unchanged.

Default | OK | Cancel | Apply

about 60% in a single blow. Further checkboxes activate the use of the SPF system, and connection to RBLs and SURBLs, with a customizable list of reputation data sources. An address whitelist is automatically populated with accepted mails within inboxes found on the system, and can be further expanded with ease. A further white/ blacklisting system, based on words and phrases, is equally easy to configure, allowing specific terms to be barred or allowed with a few clicks.

We had neither the time nor resources to test the filtering performance properly, but a quick look at it over a few days' worth of personal mails seemed to show a perfectly reasonable spam detection rate, with no false positives spotted. Even if the catch rate were to be at a bare minimum, the depth and simplicity of the user configuration options make this a remarkable and highly effective tool.



Web Protection (Teenager)

Filtering Options | Scanning Options | Define Time-Restriction | Popup Filter

Status

(✓) Active ( ) Inactive ( ) Block Web Access

Filter Categories... ● Allow ● Block

Category Name	Status	Type
Pornography	Block	Customize
Gambling	Block	Customize
Chat	Block	Customize
Alcohol	Block	Customize
Violence	Block	Customize
Drugs	Block	Customize
Ratings_block_category	Block	Customize
Websites_Allowed	Allow	Customize

Add | Del

Category: Pornography

Words / Phrases

Orgasm
18 anni
18 ans
18 jaar

Site Names

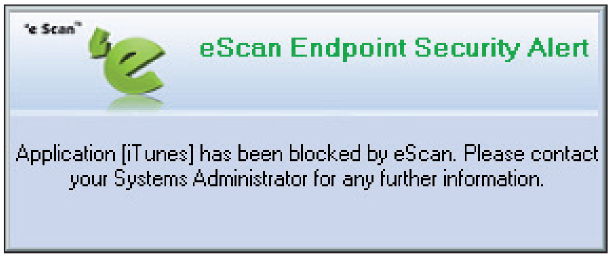
01nichons.com
01tgp.com
01tube.com
0204girls.com

Add | Del | Save

Filter Options

- [✓] Add sites rejected by the filter to Block category

OK | Cancel | Apply



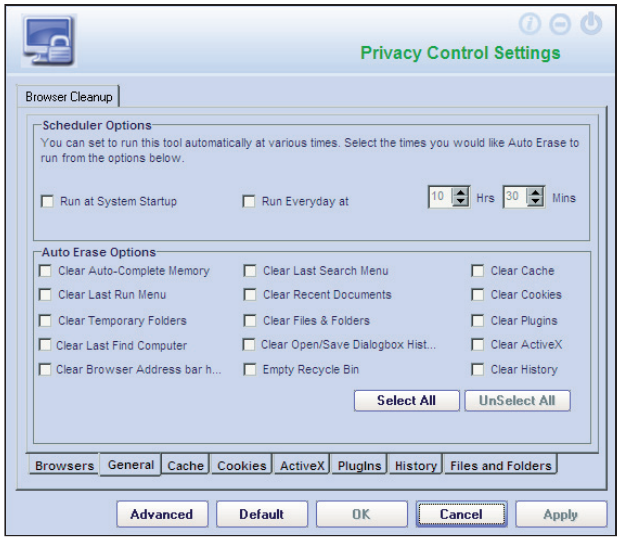
eScan Endpoint Security Alert

Application [iTunes] has been blocked by eScan. Please contact your Systems Administrator for any further information.

Moving on to the remainder of the modules under the main 'Protection' tab, the first on the list is labelled 'Web Protection'. Having at first assumed this would be a component of the anti-virus protection (scanning web downloads as they came down), we were surprised to find that it is in fact a fully featured parental control system, with four levels of control. The control levels are intended for small children, two classes of teenagers and adults. The complex and in-depth control system allows access to sites to be blocked by category, name and keywords; defaults in the 'Pornography' section include 'hot bottom' and 'legume'. A maximum permitted frequency of such shocking words is configurable for each user category. Specific content types can also be blocked, the filter using data from respected online safety agencies including *SafeSurf*. Time restrictions can also be configured to limit usage, and full logging of all violations is provided. A lengthy default whitelist of sites is provided, which is again highly configurable, and for the youngest category of users only these approved sites are accessible.

Next up is a section rather vaguely labelled 'Endpoint Security'. This is an application control system, disabled by default, with a well-stocked list of applications split into categories including games, IM and P2P applications, and media players, with further user-defined options easily set up. The second part of this module, labelled 'USB control', manages connection of USB drives and devices to the system, allowing the user to specify that a password must be entered before a drive can be mounted, to disable *Windows*' notoriously dangerous autorun set-up, to scan USB drives or mount them read-only, and even to whitelist trusted drives.

The final section of the main protection tab, 'Privacy Control', provides automated removal of browsing history and other traces. The single page of the control interface provides scheduled cleanup settings, along with the ability to browse and delete or block specific cookies, remove browser plug-ins and helper objects, peruse browsing history, purge caches and even add cache folders for purging. All of these options cover all installed browsers (with the apparent exception of *Google Chrome*). Yet again, the design is straightforward and easy to understand and use.



Privacy Control Settings

Browser Cleanup

Scheduler Options

You can set to run this tool automatically at various times. Select the times you would like Auto Erase to run from the options below.

☐ Run at System Startup ☐ Run Everyday at 10 Hrs 30 Mins

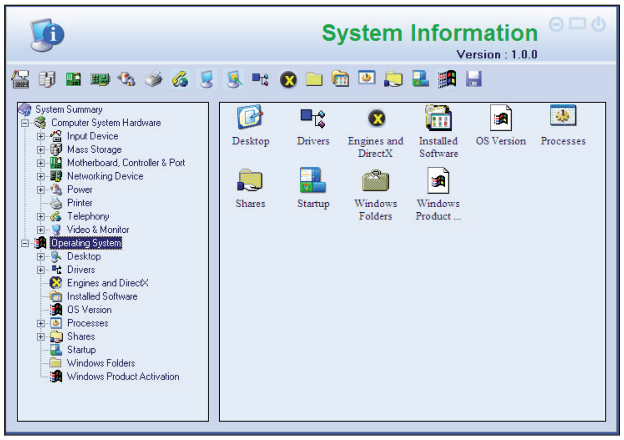
Auto Erase Options

☐ Clear Auto-Complete Memory	☐ Clear Last Search Menu	☐ Clear Cache
☐ Clear Last Run Menu	☐ Clear Recent Documents	☐ Clear Cookies
☐ Clear Temporary Folders	☐ Clear Files & Folders	☐ Clear Plugins
☐ Clear Last Find Computer	☐ Clear Open/Save Dialogbox Hist...	☐ Clear ActiveX
☐ Clear Browser Address bar h...	☐ Empty Recycle Bin	☐ Clear History

The entire set-up can be protected with an administrator password, allowing parents or owners to maintain tight control of their machine, whoever is using it.

One final section merits some comment: a tab on the main screen labelled 'Tools'. Here, the main tool is a system information monitor, which displays a vast range of data on the system being watched. Full details of hardware and drivers are covered, similar to the information provided by the standard *Windows* 'System Information' tool but considerably more user-friendly, and combined with a range of other items such as lists of installed software, shared folders and drives, startup items, and running processes. Most, if not all of the information here is available from various normal *Windows* components or free tools available from *Microsoft* (particularly since its acquisition of the *Sysinternals* product set), but having so much together in a single, simple interface is a boon for the power user.

Also listed under 'Tools' are options to report a bug or defect to the manufacturer, and to download the latest



'hotfix'– presumably managed separately from simple detection updates, possibly as system restarts may be required to get them running. Finally, there is a scanner which checks through the registry and system areas and resets any significant changes to the *Windows* defaults. This chugs through the system resetting various options, such as settings for displaying hidden system files and so on, to the *Windows* defaults.

CONCLUSIONS

Having had considerable experience of *eScan* products from numerous VB100 outings, we have always found it to be a solid and well-designed implementation of the *Kaspersky* detection engine, a pleasure to test with its stability and straightforward, sensible layout. Having expected a suite version to include the standard additions of firewall and anti-spam and little else, we were both surprised and impressed by the full range of additional tools available. The application control and parental control options, the privacy monitor and system analysis tools, the mail and web policy enforcement and much more were all unexpected delights. That just about every component seemed to work solidly and without fuss was another bonus, but the clarity and simplicity of the configuration system really puts the product into the top league. Providing such user-friendly design, making the more sophisticated areas of the product accessible to any user regardless of their technical ability without compromising on the finely graded configurability, is truly a remarkable feat.

Reading back through this review, parts of it may come across as gushing and over-enthusiastic, but I am genuinely most impressed with this suite. It seems to provide, for home users, levels of control over their systems usually only available to enterprise network admins running a range of security products and custom client lockdown scripts. I can only recommend that users, guided by what will hopefully be a decent help system when it comes on line, spend the time to plough through the vast range of options and settings dialogs to get the most out of the great degree of control available here. Even with its default set-and-forget settings, the suite provides a top-class level of protection, but with so much more to offer it would be a shame to see any of it go to waste.

Technical details

MicroWorld eScan Internet Security Suite was variously tested on:

Intel Pentium 4 1.6 GHz, 512 MB RAM, running *Microsoft Windows XP Professional SP2.*

AMD Athlon64 3800+ dual core, 1 GB RAM, running *Microsoft Windows XP Professional SP3* and *Windows Vista x64 SP1*.

Intel Atom 1.6 GHz netbook, 256 MB RAM, running *Microsoft Windows XP Professional SP3*.

END NOTES & NEWS

The FIRST Technical Colloquium on Computer Security Incident Handling 2009 takes place in Riga, Latvia, 19–21 January 2009. For details see http://www.first.org/.

Black Hat DC 2009 takes place 16–19 February 2009 in Washington, DC, USA. Online registration is now open. For details see http://www.blackhat.com/.

CanSecWest 2009 will take place 16–20 March 2009 in Vancouver, Canada. For full details including online registration and a preliminary agenda, 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/.

The Computer Forensics Show will be held 27–29 April 2009 in Washington, DC, USA. For more information see http://www.computerforensicshow.com/

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

The 3rd International CARO Workshop will take place 4–5 May 2009 in Budapest, Hungary. This year the focus of the workshop will be on the technical aspects and problems caused by exploits and vulnerabilities in the broadest sense. A call for papers has been issued (deadline 15 January). For more details see http://www.caro2009.com/.

The 18th EICAR conference will be held 11–12 May 2009 in Berlin, Germany, with the theme 'Computer virology challenges of the forthcoming years: from AV evaluation to new threat management'. For more information see http://eicar.org/conference/.

NISC 10 will take place 20–22 May 2009 in St Andrews, Scotland. For more details including provisional agenda and online registration see http://www.nisc.org.uk/.

The 21st annual FIRST conference will be held 28 June to 3 July 2009 in Kyoto, Japan. The conference focuses on issues relevant to incident response and security teams. For more details see http://conference.first.org/.

Black Hat USA 2009 will take place 25–30 July 2009 in Las Vegas, NV, USA. Training will take place 25–28 July, with the briefings on 29 and 30 July. Online registration will open in February 2009, when a call for papers will also be issued. For details see http://www.blackhat.com/.

The 18th USENIX Security Symposium will take place 12–14 August 2009 in Montreal, Canada. The 4th USENIX Workshop on Hot Topics in Security (HotSec '09) will be co-located with USENIX Security '09, taking place on 11 August. For more information see http://www.usenix.org/events/sec09/.

VB2009 will take place 23–25 September 2009 in Geneva, Switzerland. VB is currently seeking submissions from those wishing to present papers at VB2009. A full call for papers can be found at http://www.virusbtn.com/conference/vb2009/call/. For details of sponsorship opportunities and any other queries relating to VB2009, please email conference@virusbtn.com.

SUBSCRIPTION RATES

Subscription price for 1 year (12 issues):

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

Editorial enquiries, subscription enquiries, orders and payments:
Virus Bulletin Ltd, The Pentagon, Abingdon Science Park, Abingdon, Oxfordshire OX14 3YP, England
Tel: +44 (0)1235 555139 Fax: +44 (0)1865 543153
Email: editorial@virusbtn.com Web: http://www.virusbtn.com/

vb Spam supplement

CONTENTS

S1 **NEWS & EVENTS**

S1 **COMPARATIVE REVIEW – PROLOGUE**
Introducing VB anti-spam testing

NEWS & EVENTS

TWITTER-PHISH

The new year has seen yet another new angle of attack for phishers as messaging via social networking site *Twitter* became the latest way for phishers to fool victims into parting with their personal details.

Twitter users have reported being sent phishing messages which invite the recipient to log onto *Twitter* to view a particular blog or page. Of course, the link in the message directs the user to a fake *Twitter* site from which the user's login details are harvested.

The attackers have also been using the *Twitter* identities of their victims to launch a second wave of messages in which users are fooled into handing over more personal details such as mobile telephone numbers under the pretence of taking part in a prize draw. Security experts have speculated that the spammers may be earning a commission via affiliate links by directing traffic to these sites.

A warning has been added to the *Twitter* site, urging users to exercise caution when they reach what appear to be *Twitter* login pages. A blog entry about the incident also provides more detailed information about phishing.

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/.

COMPARATIVE REVIEW – PROLOGUE

INTRODUCING VB ANTI-SPAM TESTING

Martijn Grooten

According to my mother, dolphins are the best spam filter. She told me that ever since her email program was configured so that an image of dolphins was added to the bottom of every email she sent, she had stopped receiving emails about Viagra.

Unfortunately, the dolphins did not really eat the spam. As it turned out, the addition of the image to her email footers coincided with her starting to use a new email address, which of course was the real reason for the reduction in spam. And, while the dolphins are still at the bottom of each email she sends, the spam has now returned.

So what is the best spam filter? End-users have little factual information upon which to base their choice of filter, and even the vendors themselves have little idea of their products' performance compared to that of their competitors.

It seems that there is a need for independent and regular spam filter performance testing, and *Virus Bulletin* has decided to use the experience gained during more than a decade of anti-malware comparative testing to develop a regular anti-spam product test. Following months of internal discussion, culminating in an informal, yet fruitful meeting with members of the anti-spam industry at the last *VB* conference, a test methodology has been drawn up. Testing will start during the early months of 2009; this article outlines the proposed test set-up.

WHO ARE THE TESTS FOR?

While writing this article, I have received many emails from representatives of anti-spam companies politely, yet impatiently enquiring as to when we will be ready to start anti-spam testing. There certainly is big demand for product testing from within the industry: companies want to know how well they are performing compared to their competitors. If they are doing well, they want to boast about it to potential customers, while if they fail to live up to their promises, they want to know what parts of their product they need to improve. Just as *Virus Bulletin* has been running anti-malware tests for over a decade and helping AV developers improve their products, the results of *VB*'s anti-spam tests will help anti-spam vendors improve their filters.

More obviously, anti-spam customers want to know how well their chosen product, or one they are interested in purchasing, has performed in tests: whether independent sources back up the claims made by the vendors and whether those claims are based on long-term good performance.

However, a comparative test of anti-spam filters is more than just the comparison of various products: as different filters use different filtering methods, the test will implicitly compare these methods as well. This information should be valuable for the anti-spam industry as a whole.

PROBLEMS WITH ANTI-SPAM TESTING

In theory, setting up a comparative anti-spam test is easy: one sets up a forwarder that sends incoming mail to *n* mailboxes, one for each product to be tested. After a fixed period of time, one counts the number of misclassified ham messages and misclassified spam messages and thus ends up with a two-dimensional score for each product.

In practice, a multitude of problems arise, especially if one wants the anti-spam test to reflect a real-life situation. The first of these is the actual definition of spam, over which there is not more than a vague general agreement. But even assuming that one does have a satisfying definition of spam, it is not straightforward even to manually sort all incoming email according to this definition. If we at *Virus Bulletin* were able to do this with 100% accuracy in an automated way, we would likely quit our spam testing and start selling a classifier-filter commercially.

But even sidestepping the classification and definition issues, it is possible that the set-up described above will confuse spam filters. These might see an email claiming to be from joe@example.com that, from the filter's point of view, is sent from the IP address of the forwarder. Based on this information a filter might (incorrectly) decide to classify the email as spam. To deceive spammers, a filter might temporarily block an email and ask the sender to resend it after a given amount of time (a method known as greylisting); while the forwarder could be programmed to obey this request, the fact that different filters will send different 'blocking messages' means that it would never be able to do exactly what the original sender would have done.

TESTING GUIDELINES

The list of problems with possible anti-spam test set-ups is much longer than the few highlighted above. However, we do believe that it is possible to run good, representative and reliable anti-spam tests. To this end, we have set five conditions that our tests will fulfil:

- Comparative: in order to be truly comparative all products will be tested in parallel and will be sent the same or a statistically comparable email stream.
- Real-time: the emails will be sent to the products in real time, with no delay.
- Real email: all the emails used in the test will have actually been sent from external sources; no extra ham or spam will be generated to increase the test sets.
- Unbiased: when classifying emails, the testers (or anyone else involved in the classifying of emails) will have no information on how the email has been labelled by any of the test products.
- Statistically valid: there is no 'wildlist' of all the spam sent out in a given period of time and thus any test set of emails will be nothing but a sample of the billions of spam and ham emails sent during the test period. This sample should be large enough for the testers to make claims about the products' spam catch rates and their false positive rates.
- Non-isolated environment: the products being tested will be able to connect to the Internet during testing.

In our opinion, any good anti-spam test should fulfil the six conditions listed above.

However, it would be an audacious claim to state that following these guidelines will guarantee that a test will be faultless. The reason for this is deeper than the fact that any test based on statistics is bound to incorporate some error. Many spam filters block traffic at the SMTP level, based on the IP address of the sender, the content of the MAIL FROM or RCPT TO commands or a combination of these. When this happens, the email is never received and while this generally boosts the filter's performance, it is impossible for the tester to decide whether the unreceived email was spam or ham.

ANTI-SPAM TEST SET-UP

Virus Bulletin intends to test spam filters using two test set-ups per product.

1. MEASURING THE FALSE POSITIVE RATIO

The first set-up uses the existing *Virus Bulletin* email stream. An incoming SMTP server will be built to do four things:

1. Accept all email, regardless of whether the address in the RCPT TO command exists on our mail system.
2. Store the full email (including all headers), as well as the full SMTP transaction, in a database.
3. Forward the email to all the products taking part in the test (see below).
4. Forward the email to *Virus Bulletin*'s internal network.

The forwarding described in step 3 should leave the headers intact, but add an extra Received: header. Moreover, the MAIL FROM address should be changed using the Sender-Rewriting Scheme (SRS) to reflect the fact that the email is now sent from the forwarding SMTP server. Doing this should ensure that filters do not incorrectly classify an email as spam because it failed the SPF or DKIM test. (Of course, it may well be that the original email would have failed the SPF test, while the forwarded email will not; this is one of the reasons for testing using two parallel set-ups.) To ensure there is no bias towards any product, the forwarding will happen in a random order.

For each product, a script will run regularly on the server to read the product's log files and store the filter's classifications in the aforementioned database. For each incoming email, the database will contain a list of ham/spam classifications, one for each product.

In the case of many incoming emails, all filters will agree on the classification (all filters will classify the message as spam or all filters will classify it as ham). Where this is the case we will assume the classification to be correct – while this is not a guarantee that all the filters have got it right, it will greatly reduce the amount of time end-users need to spend manually classifying messages. Furthermore, given that our tests are comparative, it will not bias any of the products.

To classify the remaining bulk of emails, each end-user (members of the *Virus Bulletin* team) will be presented with a web interface displaying all the currently unclassified emails addressed to them and will be required to manually classify each as 'ham', 'spam' or 'unclassifiable'.

(It is possible that one or more of the filters taking part in the test perform so poorly that they disagree with the majority of the filters. If this proves to be the case, we will ignore the output of the poorly performing filters for the purposes of preliminary classification. This will be reported with the test results.)

2. MEASURING SPAM CATCH RATES

While the first set-up will mainly be used to derive a metric for the false positive ratio (i.e. the relative occurrence of incorrectly classified ham), a second set-up will be used to measure the products' spam catch rates.

To this end, we will use one or more large spam traps: domain names for which the incoming email is almost guaranteed to be spam. (At least in theory, there exist legitimate reasons for any address to be sent email, but if the email stream is large enough, the amount of legitimate email will be negligible.) Then, using round-robin DNS, we will distribute this mail stream equally among the products to be tested.

The products will thus communicate directly with the sending SMTP server. They will be free to do anything to check the credibility of the sender and to determine whether the email is spam: from slowing down the connection to discourage spammers from continuing with it (a method known as tarpitting), to checking the IP address against a DNS blacklist. However, they will not be allowed to block the connection temporarily and ask the sender to try again after a certain period of time (greylisting); round-robin DNS makes it unlikely that the second attempt arrives at the same SMTP server, thus potentially causing a number of problems, from an unbalanced stream to a long sequence of temporary failures that could ultimately cause the sending SMTP server to give up trying.

To measure the products' performance, all SMTP transactions will be logged and these log files will be compared with the number of emails that end up being classified as ham: all other emails will be considered to have been blocked as spam.

SPEED AND PERFORMANCE TESTS

Although the spam catch rate and the false positive rate are the most obvious metrics one would look at to compare anti-spam products, they are far from the only ways to describe a product's performance.

Therefore, although our initial focus will be on measuring spam catch and false positive rates, we hope in the future to look into other metrics too. These include both speed (how long does it take for a legitimate email to reach the user's inbox?) and performance (how much CPU does the spam filter use?), but may also include metrics such as the relative amount of spam blocked during the SMTP transaction.

Other metrics that may be published with the test results include the standard variation in a product's spam catch rate from its daily or hourly average. Or, in the event of a particularly large spam outbreak coinciding with a test, the products' response times to that outbreak.

DEFINITION OF SPAM

The general definition of spam upon which *VB* staff will base classification decisions is that of unsolicited bulk email: email sent in large quantities to users that have not explicitly given their consent to receive such email. However, this does not mean that everything else will be automatically classified as ham: we allow for the existence of a third category of 'unclassifiable' emails: ones which the recipient is unable to label as definitely ham or definitely spam. For instance, these may be messages sent to a predecessor's email address, and the recipient may have no way of knowing whether or not their predecessor had consented to receive mail from the sender. This category of emails will be removed entirely from the computation of the products' performance.

Even with this third category, we are aware that no end-user will be able to classify all the email they receive in a wholly consistent and accurate way. We believe that any inaccuracy introduced in this way will in fact reflect a real-life situation, and that the end-user's perception of a filter's performance is as important as its performance compared to any formal definition of spam.

REQUIREMENTS AND SETTINGS

To take part in *Virus Bulletin*'s anti-spam tests, products must be able to accept SMTP transfers and classify email into two categories: ham and spam. Products might have additional categories, such as 'possibly spam', and it will be decided on a product-by-product basis as to whether such categories are taken to mean ham or spam; this will always be done keeping the end-user in mind and will be reported with the test results.

Products must not send temporary failures to the sending SMTP server or, more generally, do anything that requires the sending server to reinitiate the connection.

Products must be able to log the results of their filters in such a way that testers can easily run a script that reads the relevant data from the results. Storing the email in two folders in a standard mail folder format, such as mbox, also counts as logging.

The products will be installed and configured using their default settings.

Products will be allowed to connect to the Internet at any time, to allow them to update themselves and to be able to test incoming email against live blacklists and whitelists.

Every test will start with a clean install of both the product and the operating system.

Neither end-users nor testers will report the results of the filters back to the products during testing.

AWARDS AND PRICING

Results of the tests, which we anticipate running six times per year, will be published in the *Spam Supplement* section of *Virus Bulletin* magazine (available only to *Virus Bulletin* subscribers), with a basic summary of the results available free of charge to all registered users on the *VB* website.

The best-performing products will be awarded a certification, similar to the current VB100 awards for anti-malware products. The precise criteria for obtaining these awards will be decided once we have started testing properly, but our current aim is for the best-performing 50 per cent to achieve the award.

Given the cost of testing, which needs to be performed in parallel and will require the use of a separate machine for each product, we will be charging companies to take part in our tests. Of course, included in the fee will be the right to display the award (if achieved) on the company's website and product literature. Moreover, companies taking part will be provided with feedback on their products' performance, which may include a full overview of the spam their product has missed and, after anonymizing, ham emails that were wrongly classified as spam.

We are well aware of the existence of free, open-source anti-spam products and do not wish our tests to exclude these. Therefore, developers of products that are available entirely free of charge, open-source and that contain no in-product advertising will not be charged to enter their products in our tests. (However, *VB* reserves the right to limit the number of such products in each test on a first-come-first-served basis.)

LOOKING AHEAD

We will be the first to admit that our tests will not be 100% accurate. However, we intend for our tests to be as close to reality as possible, and of course will continue to look for ways in which the methodology can be tweaked and improved.

VB welcomes readers' feedback on the proposed test methodology, as well as enquiries from developers interested in submitting their products for testing. Please direct all comments and enquiries to editor@virusbtn.com.