FEBRUARY 2009

virus BULLETIN

Fighting malware and spam

CONTENTS

ISSN 1749-7027

IN THIS ISSUE

RUNNING MACHINE

'Do media players, Sat Navs, SD cards or external hard drives make legitimate use of AutoRun?' asks Roel Schouwenberg.
page 2

CHAOS CENTRAL

Upon leaving the 25th Chaos Communication Congress, Morton Swimmer concluded that 2009 is going to be a very interesting year – but not in a good way. He provides a full round up of the research presented at the event.
page 11

VB100: RED HAT LINUX

This month sees the VB100's annual visit to the Linux platform, as well as the introduction of a brand new set of tests that will provide deeper insight into products' ability to keep up with the flood of new malware as well as their proactive detection capabilities. John Hawes has the details.
page 17

vbSpam supplement

This month: anti-spam news and events, and Martijn Grooten answers some of the common queries raised by vendors about the proposed test set-up for VB's upcoming anti-spam comparative testing.

'Do media players, Sat Navs, SD cards or external hard drives make legitimate use of AutoRun?'

Roel Schouwenberg
Kaspersky Lab, USA

TO RUN OR NOT TO RUN? THAT'S THE QUESTION

During 2008 a huge increase was observed in the amount of malware using the *Microsoft Windows* AutoRun functionality.

Previously, AutoRun was used mainly by malware coming from China targeting online games. Now, however, all sorts of malicious applications have been upgraded to include replication via AutoRun.

The current situation is reminiscent of the era of boot viruses. Some 15 years ago we faced a very similar problem with viruses spreading via floppies. Who doesn't remember buying brand new floppies only to find out they had been infected at the factory in which they were assembled? These days one might encounter AutoRun malware on a brand new MP3 player, digital picture frame, external hard drive or Sat Nav, to name but a few devices.

Boot viruses started to die out following *Microsoft*'s introduction of the *Win95/NT* operating system, on which a lot of malware failed to run. Can *Microsoft* repeat that trick once more?

Unfortunately, the situation we're facing now doesn't seem as easily solvable as the one we faced 15 years ago. The main problem is that it's much easier for a clean machine to be infected by AutoRun malware than with a boot virus. Up until *Windows XP SP1* an infected device only had to be plugged in for the malware to be run. With *XP SP2 Microsoft* changed how AutoRun was handled for USB devices, which meant that the malware would no longer be run automatically. However, accessing the device through *Explorer* still caused the malware to be run, and SP3 brought no change in this behaviour.

Realizing the error of its ways, *Microsoft* decided to handle things differently with *Vista*. In *Vista*, AutoRun does not play an active role by default for USB devices – a user has to activate the AutoRun command manually. However, while AutoRun no longer plays a significant role in *Vista*, AutoPlay is starting to play a bigger role for programs.

AutoPlay displays a prompt asking the user what it should do, with the default option being to run the program – and it's not hard to guess what the majority of users will do. AutoPlay also offers to always take the same action for software, effectively making it the same as *XP SP1* AutoRun.

While the user will no longer get infected (semi-)automatically by default, it's unlikely that users will suddenly start being smart enough not to launch malware. In all likelihood, *Microsoft* made this move to improve security, but felt it necessary still to offer the equivalent user experience.

Some brief tests with *Windows 7* showed the same results as with *Vista*, which pretty much means that nothing will change and infections via AutoRun/AutoPlay will continue.

It is obvious that *Microsoft* considers user-friendliness to be very important. But surely there are better ways to tackle the huge problem we are currently facing. Quite simply, there should be more differentiation between writable and read-only storage. The best course of action would simply be to eradicate AutoRun/AutoPlay for writable media – at least for programs.

This wouldn't cause a problem for U3 devices as they have a read-only part, nor would it be a problem for CD/DVD-ROMs. The risk of infected read-only media is not great, and certainly not when compared to writable media.

Do media players, Sat Navs, SD cards or external hard drives make legitimate use of AutoRun? External hard drives may do, but there's an easy solution for the problem of usability: simply allow AutoRun/AutoPlay for writable media only if the program is digitally signed. Any self-respecting company already signs its applications, so the issue of user-friendliness is not a big one at all.

Please, *Microsoft*, respond to this problem – it's not too late to fix it by making the necessary changes to *Windows 7*.

NEWS

DISMISSED EMPLOYEE PLEADS NOT GUILTY TO PLANTING MALWARE

A former contractor at US mortgage firm *Fannie Mae* has pleaded not guilty to charges of having planted malware on the firm's computer systems.

Rajendrasinh Babubhai Makwana had been working as a computer programmer at *Fannie Mae* until his contract was terminated in October 2008. Prosecutors allege that following his termination he planted malicious code on the company's systems. The code, which was embedded within a legitimate script, was designed to trigger on 31 January, overwriting data across the company's network of 4,000 servers, however it was discovered by another Unix engineer five days after Makwana's dismissal.

According to an FBI sworn statement Makwana's access to the *Fannie Mae* computer systems was not terminated immediately following his dismissal – although notified of the termination of his contract between 1 p.m. and 1.30 p.m., he was not required to turn in all of his computer equipment until the end of the day, and his access to the company's computer systems was not terminated until late in the evening. Had the malicious script not been discovered, it is estimated that the damage would have run to millions of dollars.

A trial date is expected to be set later this month. If convicted, Makwana faces a maximum sentence of 10 years in prison.

GOOGLE CONFESSES TO HUMAN ERROR

Google has apologised for a brief period during which the search engine labelled every site on the Internet as potentially dangerous. Under normal circumstances the message 'This site may harm your computer' appears next to *Google* search results if a site is known to be malicious – a blacklist providing the relevant information. However, on 31 January, for an approximately 40-minute period, all search results appeared with the warning.

According to *Google* the error occurred in the manual updating of the blacklist when the URL '/' was accidentally added to the file. The error was discovered quickly and the file was rolled back. The search engine's staggered system of updates meant that the errors began appearing between 6:27 a.m. and 6:40 a.m. PST and began disappearing between 7:10 a.m. and 7:25 a.m. PST, thus the problem lasted no longer than approximately 40 minutes for any particular user.

A statement from *Google*'s VP, Search Products & User Experience assured users that the incident would be investigated carefully and that robust file checks would be put in place to prevent it from happening again.

Prevalence Table – December 2008

Malware	Type	%
NetSky	Worm	14.57%
Invoice	Trojan	13.05%
Agent	Trojan	11.91%
Mytob	Worm	7.44%
Virut	Virus	5.94%
Suspect packers	Misc	5.71%
Zbot	Trojan	4.39%
Autorun	Worm	3.99%
Small	Trojan	3.96%
Inject	Trojan	3.05%
Dropper-misc	Trojan	3.02%
Basine	Trojan	2.74%
Mydoom	Worm	2.60%
Downloader-misc	Trojan	2.27%
Iframe	Exploit	2.23%
Bagle	Worm	1.44%
Lineage/Magania	Trojan	1.38%
Zafi	Worm	1.32%
Mdrop	Trojan	1.16%
Hupigon	Trojan	1.10%
OnlineGames	Trojan	1.08%
Murlo	Trojan	0.81%
Delf	Trojan	0.75%
PWS-misc	Trojan	0.61%
Sality	Virus	0.50%
Alman	Worm	0.43%
Grew	Worm	0.26%
Cutwail/Pandex/Pushdo	Trojan	0.25%
Womble	Worm	0.22%
Klez	Worm	0.20%
Heuristic/generic	Virus/worm	0.12%
Areses/Scano	Worm	0.11%
Mabutu	Worm	0.10%
Others[1]		1.29%
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 THREE

Peter Ferrie
Microsoft, USA

New anti-unpacking tricks continue to be developed as the older ones are constantly being defeated. This series of articles (see also [1, 2]) describes some tricks that might become common in the future, along with some countermeasures.

This article will concentrate on anti-debugging tricks. All of these techniques were discovered and developed by the author of this paper.

1. Miscellaneous tricks

1.1 Ctrl-C

When a user presses the Ctrl-C key combination while a console window has the focus, *Windows* calls the kernel32 IsDebuggerPresent() function and issues the DBG_CONTROL_C (0x40010005) exception if the function returns true. This exception can be intercepted by an exception handler or an event handler, but as noted previously [2], the exception might be consumed by a debugger instead. As a result, the absence of the exception can be used to infer the presence of a debugger. The application can register an exception handler in the usual way – SEH, VEH, SafeSEH – or register an event handler by calling the kernel32 SetConsoleCtrlHandler() function. The exception can then be forced to occur by calling the kernel32 GenerateConsoleCtrlEvent() function.

1.2 Ctrl-break

Similarly, when a user presses the Ctrl-break key combination while a console window has the focus, *Windows* calls the kernel32 IsDebuggerPresent() function and issues the DBG_CONTROL_BREAK (0x40010008) exception if the function returns true. As above, this exception can be intercepted by an exception handler, but the exception might be consumed by a debugger instead. Once again, the absence of the exception can be used to infer the presence of a debugger. The application can register an exception handler in the usual way or register an event handler by calling the kernel32 SetConsoleCtrlHandler() function. The exception can then be forced to occur by calling the kernel32 GenerateConsoleCtrlEvent() function.

1.3 Interrupt 0x2D

Whenever a software interrupt exception occurs, the exception address and the EIP register value point to the next instruction that will execute – which is after the instruction that caused the exception. A breakpoint exception is treated as a special case. When an EXCEPTION_BREAKPOINT (0x80000003) exception occurs, *Windows* assumes that it has been caused by the 'CC' opcode ('INT 3' instruction) and decreases the exception address by one before passing the exception to the exception handler. The EIP register value is not affected. Thus, if the 'CD 03' opcode (long form 'INT 03' instruction) is used, the exception address will point to '03' when the exception handler receives control.

However, when interrupt 0x2D is executed, *Windows* uses the current EIP register value as the exception address and increases the EIP register value by one. Finally, it issues an EXCEPTION_BREAKPOINT (0x80000003) exception. Thus, if the 'CD 2D' opcode ('INT 0x2D' instruction) is used, the exception address points to the instruction immediately following the interrupt 0x2D instruction, as for other interrupts, and the EIP register value points to a memory location that is one byte after that.

After an exception has occurred, and in the absence of a debugger, execution will resume by default at the exception address. The assumption is that the cause of the exception will have been corrected, and the faulting instruction will now succeed. In the presence of a debugger, and if the debugger consumed the exception, execution will resume at the current EIP register value.

The interrupt 0x2D behaviour can be troublesome for debuggers. The problem is that the EIP register value points to a position one byte after the location at which execution is 'expected' to resume. This can result in a single-byte instruction being skipped, or the execution of a completely different instruction because the first byte is missing. These behaviours can be used to infer the presence of the debugger.

Example code looks like this:

```
    xor  eax, eax ;set Z flag
    push offset l1
    push d fs:[eax]
    mov  fs:[eax], esp
    int  2dh
    inc  eax ;debugger might skip
    je   being_debugged
    ...
l1: xor al, al
    ret
```

This behaviour has been documented [3], but apparently is not fully understood. *Turbo Debug32* is one debugger that is not affected, because it always decreases the EIP register value when an EXCEPTION_BREAKPOINT (0x80000003) exception occurs. In contrast, *OllyDbg*

adjusts the EIP register value according to the instruction that appears at the exception address. If a 'CC' opcode ('INT 3' instruction) is seen, then *OllyDbg* will reduce the EIP register value by one; if the 'CD 03' opcode (long form 'INT 03' instruction) is seen, then *OllyDbg* will reduce the EIP register value by two, in order to be able to step correctly over the instruction (this can be used to detect the presence of *OllyDbg*, based on the instruction that is executed next). If neither opcode is seen, then the EIP register value is not altered. The result for *OllyDbg* is that one byte is skipped.

1.4 Interrupt 0x41

Interrupt 0x41 can display different behaviour depending on whether or not a kernel-mode debugger is present. This interrupt descriptor normally has a descriptor privilege level (DPL) of zero, which means that it cannot be issued from ring 3. Attempts to execute this interrupt directly result in a general protection fault (interrupt 0x0D) being issued by the CPU, eventually resulting in an EXCEPTION_ACCESS_VIOLATION (0xC0000005) exception. However, some debuggers hook interrupt 0x41 and adjust the DPL to three, so that they can be called from user-mode code.

1.5 Missing exceptions

Heap and resource functions, among others, can be forced to cause a debug break. What they have in common is a check of the PEB->BeingDebugged flag. The presence of the debugger can be faked in order to force the interrupt 3 exception to occur. The absence of the exception is a sign that a real debugger intercepted it.

Example code looks like this:

```
    xor  eax, eax
    push offset l1
    push d fs:[eax]
    mov  fs:[eax], esp
    mov  eax, fs:[30h]
    inc  b [eax+2]
    push offset l2
    call HeapDestroy
    jmp  being_debugged
l1: ...
;HEAP_VALIDATE_PARAMETERS_ENABLED
l2: dd 0, 0, 0, 40000000h
```

2. SoftICE-specific

For many years, *SoftICE* was the most popular debugger available for the *Windows* platform (development of the program was discontinued in 2006). *SoftICE* is a debugger that makes use of a kernel-mode driver in order to support the debugging of both user-mode and kernel-mode code, including transitions in either direction between the two. It has a number of vulnerabilities.

2.1 Interrupt 3

SoftICE contains a 'backdoor' interface on interrupt 3. It is accessed by setting the value of the SI register to 'FG', and the DI register to 'JM'. These values are the initials of the authors of the original *SoftICE for DOS*: Frank Grossman and Jim Moskun. The AH register contains the function index. The allowed values are 00, 09, 0x80-0x83 and 0xA0. Other registers have meaning, depending on the function that is called. This interface has existed since the DOS version, so it is well known, but it remains poorly documented. The lack of documentation might be the reason why the interface has not been investigated closely. However, if it had been, then several vulnerabilities might have been discovered and corrected. These vulnerabilities allow for multiple denial-of-service attacks. Two major function indexes are vulnerable. They are 09 and 0x83.

The function index 09 uses the AL register to select a subfunction. The allowed values are 00, 02-05, 07-0x0E, 0x18 and 0x19. Of these, all of the subfunctions are vulnerable apart from 00, 0x18 and 0x19. In the case of subfunctions 02 and 03, the EBX register is the trigger. For the others, the EDX register is the trigger, with the ECX contributing a length which must not exceed a certain value.

Example code looks like this:

```
mov ax, 907h
xor ecx, ecx
xor edx, edx
mov si, "FG"
mov di, "JM"
int 3
```

For the function index 0x83, the BX register is the trigger. This register is used as a pointer to memory but, unless allocated explicitly by calling NtAllocateVirtualMemory() directly, it will always point to inaccessible memory.

Example code looks like this:

```
mov ah, 83h
xor ebx, ebx
mov si, "FG"
mov di, "JM"
int 3
```

2.2 Interrupt 0x41

As noted above, some debuggers hook interrupt 0x41 and adjust the DPL value to three, so that they can be called from user-mode code. *SoftICE* is one of those debuggers. However, the changes are not visible from within *SoftICE* – the IDT command shows the original interrupt 0x41 handler address with a DPL of zero.

The interrupt 0x41 interface includes a debugger installation check. This is demonstrated by calling interrupt 0x41 with AX=0x004F. It returns AX=0xF386 if the debugger is present. This interface might look familiar – a similar one existed for DOS on interrupt 0x68, where passing AX=0x4300 returned AX=0xF386 if a debugger was present. Some anti-unpacking routines still use the interrupt 0x68 interface, even though it is not supported on *Windows NT*-based platforms.

SoftICE exposes a large interface on interrupt 0x41. The interface supports functions to emit debug and error messages, create and destroy segments, and load and unload DLLs. Unfortunately, careless coding allows multiple opportunities for denial-of-service attacks. One example is OutputDebugString. There are 16- and 32-bit versions of this function. Both of them are vulnerable. These functions accept a pointer in the [E]SI register to the string to print. *SoftICE* probes the memory to ensure that the string can be read, but the probe covers only the first character. After that, the memory is accessed blindly, and if the string crosses into an unmapped region, then *SoftICE* will cause a kernel-mode crash (blue screen).

Example code for OutputDebugString looks like this:

```
push 1
mov  ecx, esp
push 4 ;PAGE_READWRITE
;MEM_COMMIT + MEM_RESERVE
push 3000h
push ecx
push 0
push ecx
push -1 ;GetCurrentProcess()
call NtAllocateVirtualMemory
;OutputDebugString
mov  al, 12h
mov  esi, 0fffh
mov  [esi], al ;non-zero
int  41h
```

Example code for OutputDebugString32 looks like this:

```
xor  esi, esi
push 4 ;PAGE_READWRITE
push 1000h ;MEM_COMMIT
push 1
push esi
call VirtualAlloc
add  ax, 0fffh
xchg esi, eax
;OutputDebugString32
mov  al, 2
mov  [esi], al ;non-zero
int  41h
```

Another example in OutputDebugString32 is 'BCHKW' in a read-only page. When *SoftICE* sees 'BCHKW' anywhere within the string, it attempts to overwrite the first byte of the string with a zero, without checking if the page is writable.

Example code looks like this:

```
;OutputDebugString32
push 2
pop  eax
mov  esi, offset l1
int  41h
...
l1: db "BCHKW"
```

2.3 DeviceIoControl

If *SoftICE* is installed, then the DbgMsg.sys driver is loaded, even if *SoftICE* isn't running. DbgMsg.sys exposes an interface that can be called via the kernel32 DeviceIoControl() function. That code contains several vulnerabilities. For example:

```
   mov  edx, [ebp+arg_0]
   mov  eax, [edx+60h]
   mov  ecx, [eax+0Ch]
   sub  ecx, 222007h
   jz   l1
   ...
l1: mov ecx, [edx+3Ch]
   ...
   push ecx
   call l2
   ...
l2: ...
   mov  ebx, [ebp+arg_0]
   and  byte ptr [ebx], 0 ;bug here
```

The write to [ebx] without checking first if the pointer is valid leads to a kernel-mode crash (blue screen) if the output buffer parameter is invalid or read-only.

Example code looks like this:

```
xor   ebx, ebx
push  ebx
push  ebx
push  3 ;OPEN_EXISTING
push  ebx
push  ebx
push  ebx
push  offset l1
call  CreateFileA
push  ebx
push  ebx
push  ebx
push  ebx
push  1 ;non-zero
push  ebx
push  ebx
push  222007h
push  eax
call  DeviceIoControl
...
l1: db "\\.\NDBGMSG.VXD", 0
```

There is another vulnerability in the following code:

```
   mov  edx, [ebp+arg_0]
   mov  eax, [edx+60h]
   mov  ecx, [eax+0Ch]
   sub  ecx, 222007h
   ...
   push 4
   pop  esi
   sub  ecx, esi
   ...
   sub  ecx, esi
   jz   l1
    ...
l1: mov esi, [edx+3Ch]
   ...
   mov  [esi], eax ;bug here
```

The write to [esi] without checking first if the pointer is valid leads to a kernel-mode crash (blue screen) if the output buffer parameter is invalid or read-only.

Example code looks like this:

```
xor ebx, ebx
push ebx
push ebx
push 3 ;OPEN_EXISTING
push ebx
push ebx
push ebx
push offset l1
call CreateFileA
push ebx
push ebx
push ebx
push ebx
push ebx
push ebx
push ebx
push 22200fh
push eax
call DeviceIoControl
...
l1: db "\\.\NDBGMSG.VXD", 0
```

There is yet another vulnerability in the following code:

```
   mov  edx, [ebp+arg_0]
   mov  eax, [edx+60h]
   mov  ecx, [eax+0Ch]
   sub  ecx, 222007h
   ...
   push 4
   pop  esi
   sub  ecx, esi
   ...
   sub  ecx, esi
   ...
   sub  ecx, esi
   ...
   sub  ecx, esi
   jz   l1
   ...
l1: mov esi, [eax+10h]
   ...
   cmp  [esi], ebx ;bug here
```

This time the read from [esi] without checking first if the pointer is valid leads to a kernel-mode crash (blue screen) if the input buffer parameter is invalid.

Example code looks like this:

```
xor  ebx, ebx
push ebx
push ebx
push 3 ;OPEN_EXISTING
push ebx
push ebx
push ebx
push offset l1
call CreateFileA
push ebx
push ebx
push ebx
push ebx
push ebx
push ebx
push 1 ;non-zero
push 222017h
push eax
call DeviceIoControl
...
l1: db "\\.\NDBGMSG.VXD", 0
```

SoftICE also supports the writing of certain values to arbitrary memory locations. The arbitrary writing is possible because *SoftICE* does not perform sufficient address validation. While the values that can be written might seem to be of little interest – primarily the value zero – they can form part of a multi-stage attack. For example, by writing a zero to a particular location, a conditional branch can be turned into a do-nothing instruction. When applied to system-sensitive code, such as the granting of privileges, the result of such a modification allows even the least privileged account to bypass all system protection.

2.4 Fake section table

SoftICE contains an incorrect method for calculating the location of the section table. The problem is that *SoftICE* relies on the value in the PE->NumberOfRvaAndSizes field to determine the size of the optional header instead of using the value in the PE->COFF->SizeOfOptionalHeader field. As a result, it is possible to create a file with two section tables, one that *SoftICE* sees, and one that *Windows* sees. Fortunately, that does not seem to provide any scope for malicious use.

2.5 Section table placement

However, *SoftICE* also contains an off-by-one vulnerability when checking whether the section table that it sees resides

wholly within the file. Specifically, *SoftICE* wants to access the last byte of that section table in order to force in the page. However, because of an incorrect bounds check, *SoftICE* can be coerced into accessing one byte beyond the end of the section table.

Of course, a file can be created with no purely virtual sections, and the PE header and section table can be moved to the end of the file, but there is no need to go to such trouble. Due to the bug described above, it is possible to alter only the PE->NumberOfRvaAndSizes value to make the section table appear anywhere in the file, including at the very end. If the section table ends exactly at the end of the image, then when *SoftICE* accesses the one byte beyond the end of that table, a page fault will occur at a critical point and completely destabilize the system. This bug was introduced during an attempt to fix an earlier bug [4], whereby no checking at all was done prior to accessing memory.

2.6 Device names

It is interesting that we continue to see CreateFile('\\.\NTICE') in malware, given that *SoftICE* v4.x does not create a device with such a name in *Windows NT*-based platforms.

Instead, the device name is '\\.\NTICExxxx'), where 'xxxx' is four hexadecimal characters. The characters are the 9th, 7th, 5th and 3rd characters from the data in the 'Serial' registry value. This value appears in multiple places in the registry. The *SoftICE* driver uses the 'HKLM\System\CurrentContrlSet\Services\NTice\Serial' registry value. The nmtrans DevIO_ConnectToSoftICE() function uses the 'HKLM\Software\NuMega\SoftIce\Serial' registry value. The algorithm that *SoftICE* uses to obtain the characters reverses the string, then, beginning with the third character, takes every second character for four characters. There is a simpler method to achieve this, of course.

Example code looks like this:

```
  xor   ebx, ebx
  push  eax
  push  esp
  push  1 ;KEY_QUERY_VALUE
  push  ebx
  push  offset l2
  push  80000002h ;HKLM
  call  RegOpenKeyExA
  pop   ecx
  push  0dh ;sizeof(l3)
  push  esp
  mov   esi, offset l3
  push  esi
  push  eax ;REG_NONE
  push  eax
  push  offset l4
  push  ecx
  call  RegQueryValueExA
  push  4
  pop   ecx
  mov   edi, offset l6
l1: mov al, [ecx*2+esi+1]
  stosb
  loop  l1
  push  ebx
  push  ebx
  push  3 ;OPEN_EXISTING
  push  ebx
  push  ebx
  push  ebx
  push  offset l5
  call  CreateFileA
  inc   eax
  jne   being_debugged
  ...
l2: db  "Software\NuMega\SoftIce", 0
l3: db  0dh dup (?)
l4: db  "Serial", 0
l5: db  "\\.\ntice"
l6: db  "xxxx", 0
```

3. SoftICE extensions

SoftICE supports plug-ins to some degree. Many packers have been written to detect *SoftICE*, so one plug-in (so far) has been written to attempt to hide *SoftICE* from those packers.

3.1 ICEExt

ICEExt fixes a bug in earlier versions of the ntice _chkstk() function. The function previously used the wrong register to set the new stack pointer, resulting in a kernel-mode crash (blue screen) if the function was ever called.

ICEExt hooks the ntoskrnl ZwCreateFile() function directly in the Service Descriptor Table. The hook examines the specified filename, and then denies access to the 'NTICE', 'SIWVIDSTART' and 'SIWSYM' device names. The comparison is case-insensitive and uses a maximum length, so it will also protect the newer 'NTICExxxx' device name correctly. However, there is a bug in the code, which does not check whether the ObjectAttributes->ObjectName parameter points to a valid memory address. An invalid memory address causes a kernel-mode crash.

ICEExt hooks the ntoskrnl ZwQuerySystemInformation() function directly in the Service Descriptor Table. The hook calls the original ntdll ZwQuerySystemInformation() function, and then checks whether the SystemInformationClass is the SystemModuleInformation class. If it is, then *ICEExt* searches the returned module list and replaces the first reference to 'NTICE.SYS' with 'TROF2.SYS'.

ICEExt hooks the ntoskrnl ZwQueryDirectoryObject() function directly in the Service Descriptor Table. This function was introduced in *Windows 2000*. *ICEExt* calls the original function, and then replaces with 'SSINF' any driver type whose name is 'NTICE'. However, there is a bug in this code, which does not check whether the buffer parameter points to a valid memory address. An invalid memory address causes a kernel-mode crash.

ICEExt hooks interrupt 3 directly in the Interrupt Descriptor Table. The hook denies access to the *SoftICE* 'backdoor' interface.

ICEExt restores to zero the DPL of the interrupt 1 and interrupt 0x41 descriptors.

The author of *ICEExt* has yet to respond to the report.

3.2 SoftICE Cover

SoftICE Cover is a tweaked version of *SoftICE*. It runs only on *Windows XP*, and installs the final version of *SoftICE*. It allows the user to select which characteristics to hide. There are several options. The video and core drivers can be renamed, the interrupt 3 hook can be disabled, and the 'FGJM' and 'BCHK' interfaces can be 'disabled' (they are replaced by random alphabetic characters). However, this does not protect *SoftICE* against the other attacks, such as the malformed file denial of service.

In part four of this series next month we will look at anti-debugging tricks that target the *Syser* debugger – a lesser-known debugger that might be considered to be a successor to *SoftICE*, since it can run on *Windows Vista*.

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. http://www.virusbtn.com/pdf/magazine/2008/200812.pdf.

[2] Ferrie, P. Anti-unpacker tricks – part two. Virus Bulletin, January 2009, p.4. http://www.virusbtn.com/pdf/magazine/2009/200901.pdf.

[3] http://www.openrce.org/reference_library/anti_reversing_view/34/INT%202D%20Debugger%20Detection/.

[4] http://www.honeynet.org/scans/scan33/nico/index.html#1.

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.

OPINION

IT'S TIME FOR A CHANGE

James Wolfe
Independent researcher, USA

When I first became a hobbyist in anti-virus almost two decades ago, boot sector infectors were all the rage. Lately, comparisons have been drawn between the current USB-propagated infectors and boot sector infectors (see p.2) – but while there are some similarities they are mostly only superficial.

What is much more serious is the potential for system infection and data exfiltration. A big concern is the potential for data exfiltration to be perpetrated by an insider rather than via an external smash-and-grab. As an example of how easy it can be to exfiltrate a very large amount of data, it is now common to find USB sticks that can hold a full gigabyte of data, but which measure only 34mm x 12mm x 2mm – extremely easy to hide from standard physical security screening procedures. We need innovation from within the industry so that we can provide sensible protection for our customers' data.

A few years ago I began to research the terrorism threat. I was halfway through writing a graduate-level course on the subject for a local university when it became necessary to dust off my virus research chapeau. The threat profile had changed. Whereas previously we had been up against the lone wolf, or bored troll just goofing around, we now hear reports that professional criminals working in organized groups, and even nation states are behind the scenes trying to compromise and/or steal data. How can we battle against that? Well, we certainly can't depend on our current models of protection.

For years we had it easy in this industry. A new sample would be detected on a couple of systems somewhere in the world, it would be submitted to the anti-virus labs where a new signature would be produced within a week. The outbreak would blow up into a worldwide event and we would ride to the rescue with the new signature. Today, with the advanced persistent threats and targeted malware, a single new sample on a single computer can cause a customer irrevocable damage. There is no longer any consideration for the number of samples seen prior to releasing an update. Once again, we must look for the sensible way to protect the data.

WHAT'S THE PROBLEM?

What can we do to combat these new threats? Are the new threats even the real problem? Certainly, there are lots of possible solutions but I think we should address some deep-seated issues within the industry first.

Many of us in this industry pride ourselves on being researchers, so why aren't we actually researching? Why aren't we using our massive collective intelligence to out-think the criminals? In my mind, research implies advanced modelling and a forward-looking mentality. Remember the scientific model from your school days? Within that model was a problem statement and a hypothesis. Those are the first two steps in research – I haven't seen a lot of that lately in this industry and it should be the most important part.

Why is it that, in much of the industry, R&D (research and development) has given way to M&D (marketing and development)? I understand the need to advertise, but why are we allowing the marketing departments to have input into what is produced? The industry started with some incredibly brilliant individuals who wrote programs to prevent the viruses of the day from interrupting a computer's operation. Those innovators took their programs to market but kept firm control over how they were maintained. These days it almost seems as if the business model is for the non-technical departments to tell the technical developers how the program should work – and as a result a lot of unnecessary garbage is included.

I think it is time to admit that the anti-virus programs that we use today are dead. We cannot use the technology and methodologies of the 1990s and continue to be effective. It doesn't work for the customer and (with thousands of samples being seen every day) it doesn't work for the industry. No matter how many people are hired we can't keep up – and who would want 100 megabyte (or more) daily signature updates anyway? The situation will only get worse unless we move back to the mainframe/terminal model. If the core players in the industry don't do it, then someone outside the industry will come up with an innovation that will change the players in the industry cataclysmically.

CONCLUSION

Today, the threat environment largely drives how we respond to new security issues. This is a poor operational model. If we are going to continue to tout ourselves as research scientists then we need to use the scientific model, and nowhere within that model will you find marketing. Change is coming, and living in the past by using outdated tools and methodologies is not only doing our customers a disservice, but is a one-way ticket to extinction.

Where does this leave the anti-virus industry as a whole? Well that's for us to decide. Certainly any new approach we adopt needs to focus on innovation, research, and real proactive protection. It doesn't mean the end of our industry, just a new way to do business. For years our adversaries have changed their methodologies to avoid us, have we ever really thought about changing ours? I think that now is the time for that change.

CONFERENCE REPORT

CCC 25C3

Dr Morton Swimmer
Trend Micro, USA

The Chaos Computer Club (CCC) is a German-based hacking and technology activist group. 25C3, which took place 27–30 December, was the 25th Chaos Communication Congress and my 18th year of attending the event. The congress recently expanded from three to four days, but this year it reached the capacity of its current venue, the Berlin Congress Center. The question now is how the congress will manage its expansion given that, for privacy reasons, tickets are not available for pre-purchase. Although organized by the CCC, other clubs and groups also attend, and the congress is also used as a venue for open-source project meetings and similar events.

As a consequence of the event being full this year it was harder than usual to get a seat (or even standing room) at many of the presentations. Helpfully, the organizers stream and record nearly all of the sessions, so with a working network connection you can watch a live presentation from wherever you happen to be, or watch it later by downloading the recording. I have organized this report roughly by theme, starting with mobile, then Internet security, followed by malware-specific issues, finishing with a number of miscellaneous subjects.

MOBILE SECURITY

The mobile platform continues to intrigue hackers as the technology becomes better and smart phones become more widespread. While the attention devoted to this platform is worrying, its security is being eroded only very slowly and it is impossible to know when the cataclysmic event will occur that will cause the phone vendors and the service providers finally to realize that they need to take action. Having said that, the first presentation in this category did give me some cause for concern.

Rogue phone networks

We have long considered mobile phone systems a fairly closed issue, but this may be changing. Dieter Spaar and Harald Welte demonstrated a GSM base station they had bought on *eBay*. The pair had been able to reverse engineer the base station well enough to run it as a mini-GSM network. During the talk (and the previous night) it popped up as '001 01' network and the researchers were able to make voice and SMS connections. Reverse engineering the base station and creating a system for connecting to it was certainly a tidy piece of work, but it is not clear to me whether or how quickly this could evolve into the threat of rogue phone networks. Ordinarily, a GSM phone will not connect to an arbitrary network if its home network is available, but in the US network coverage is patchy. A coverage gap could be filled by a rogue network to execute a man-in-the-middle attack or just to collect IMSI and IMEI data.

This isn't as far-fetched as it may seem – in the distant past receivers were used on analogue phone networks to sniff phone credentials. Until now, obtaining the required hardware and then understanding it well enough to be able to create a GSM network node has been a hurdle, but Spaar and Welte have now demonstrated that the hurdle can be cleared.

They also mentioned in passing a similar project that is attempting to implement GSM protocol on the GNU radio/USRP (Universal Software Radio Peripheral). If this succeeds, it will be much easier to obtain the necessary hardware, as a GNU radio (which is basically a radio receiver implemented heavily in software and capable of receiving all bands) is far more easily obtained than a GSM base station (although still costly at around EUR 1,000). The authors had bought the entire stock of base stations available on *eBay* at that time and were offering them at cost to interested parties. The GNU radio and base station projects should complement each other well as one deals with the physical layer while the other with the network link layer.

The authors' base station did not support UMTS, so they have not yet done any work in that field. With the rise of smart phones, UMTS is the protocol that is the most interesting from the Internet security perspective. UMTS does support a much more sound encryption protocol than GSM, and the encryption goes to the telco rather than stopping at the base station, so it will be much harder to find useful hacks. However, as this proof of concept has shown, it is likely only to be a matter of time.

Symbian platform

Collin Mulliner of the Fraunhofer-Gesellschaft Institute presented his findings on *Symbian OS 9* security. He chose to focus on this version as previous versions of the OS didn't have notable security mechanisms in place and the API had become more or less POSIX-compliant with this version. The *Symbian* OS runs on approximately 50% of the world's smart phones as the PDA-side operating system. Mulliner was interested in the susceptibility of the OS and its applications to buffer overflow attacks and the usefulness of the capability-based security system.

The OS itself has no buffer overflow protection mechanisms in place, such as stack canaries, but will use the code execution protection mechanisms of the CPU if they exist. For that reason Mulliner found that CPUs with ARM architecture version 5 and below were vulnerable

to attack, but version 6 and above were not. He intends to test other types of code injection attack against these newer architectures in the future. He showed the results of some of his experimentation with buffer overflows, such as making a call from shellcode and an IMEI reader. To make shellcode generation easier for this platform he created a shellcode encoder so as to avoid zero bytes and problems with the instruction cache. Finding exploitable code was done by attaching the (now free) remote debugger and fuzzing. An alternative he mentioned was using *IDA Pro*'s debugger for *Symbian*.

Mulliner also discovered what he believes is a weakness in the capabilities system of *Symbian* OS. By granting network access to virtually all applications, the shellcoder can use networking to take the available IMEI to the open code-signing site that *Symbian* provides for developers and have code signed specifically for that phone. The captcha system on that site is weak and can be broken by numerous anti-captcha services. Thus an attacker can target a mobile phone via a buffer overflow attack and then create malware for that phone and upload it. Since users are already preconditioned to press the 'Yes' button as many times as the phone asks them, he believes the success rate of such malware will be high.

iPhone

This talk was basically a deconstruction of the *iPhone*'s hardware based on the work the authors did while jail-breaking it (removing the phone's protection mechanisms). The phone is divided – like most, though not all, smart phones – into two halves: the phone and the PDA part.

The phone part deals only with making calls and transferring data over the GSM/UMTS network. It is based on the *Nucleus* OS and uses encryption extensively. It has a code trust chain that starts when the initial SIMloader is run, but the bootloader which comes next has exploitable bugs – which the authors found via hardware fuzzing and allowed them to break the trust chain. While no applications can be installed on the phone side of the *iPhone*, unlocking it is the only way to run the *iPhone* on networks other than the one for which it was configured.

The PDA part is based on *Mac OS X* with an XNU kernel. There is a nearly complete chain of code trust, with the bootloader being the exception, so this part is what needs to be broken. The phone is pretty well secured if it isn't jail-broken, but inevitably someone will find a vulnerability and there is currently no official way of writing software for the kernel of the *iPhone* (or the phone part), so there is no way of easily protecting the phone with third-party software.

I was a bit disappointed that the authors didn't go into more detail about the operating system. However, the fact that the OS is based very strongly on *Mac OS X* is probably enough to understand how it is built. (Later, I was told that the main differences from *Mac OS X* are in the GUI model, which is far cleaner and tailored to small devices.)

DECT

A group of researchers from Germany and Luxembourg presented their findings on DECT phones. Although used in very few wireless phones in the US, DECT is one of the most popular standards for wireless home phones and other short-range digital wireless devices in Europe and other parts of the world.

Communications are scrambled and presumably time-sliced in an unpredictable manner. Initially, the researchers planned to use a GNU radio/USRP to build a sniffer for DECT wireless traffic, but then stumbled upon a cheaper solution in the form of a PCMCIA card. After reverse engineering that card they were able to create a DECT sniffer for about EUR 23, which was also able to transmit (in contrast to the USRP solution where real-time transmission is hampered by the delay in processing the incoming signals).

Like GSM, the cipher used by DECT is kept secret, but security through obscurity will eventually fail unless the security is very sound. Some parts of the cipher are documented in a patent, but others are not because the encryption is implemented in a dedicated chip, and the authors had to resort to chip reverse engineering.

The end result is that with a fairly small investment in hardware, the researchers were able to eavesdrop on DECT calls, but also sniff traffic from other devices that use this system, such as electronic payment terminals and traffic lights. Eventually they will be able to associate with any given DECT network and place calls.

INTERNET SECURITY

Internet security is always a big topic. This time a few of the talks broached fairly advanced themes.

DNS mess

In July 2008, multiple vendors issued patches to their DNS servers and clients to fix a problem that had been discovered many years ago, but which was only recently recognized as severe. Like much of the Internet, DNS was designed with a threat model more focused on availability than security and is brilliant at distributing data in a federated manner. For that reason, we have come to rely on DNS to federate all sorts of data that probably shouldn't have been entrusted to a relatively insecure protocol. The bug in question allowed a

type of DNS cache poisoning due to the fact that the source ports of the DNS server were predictable.

Dan Kaminsky took us on a tour of all that he had been doing last year getting DNS patched in a coordinated way so that no vendors would prove to be in a liable position. The good news is that it is estimated that over 75% of the DNS servers were patched after one month, but there are still plenty that are not. Achieving this was no easy feat as it required the vendors to be persuaded of the severity of the problem and of the need to fix it pre-emptively. It was a good tale of setting aside egos for the greater good.

Kaminsky put forward a very strong case for getting DNSSEC into a usable state, as he grudgingly believes that it is the only way of making DNS trustworthy in the long run. Dan's thesis is that authentication on the Internet is basically broken and fixing DNS would go a long way towards re-establishing trust. DNS itself may not, ultimately, be fixable so Dan believes that fixing and then deploying DNSSEC is the only way forward.

Debian RNG

Luciano Bello and Maximiliano Bertacchini reported on a bug that was introduced into the Debian version of the random number generator (RNG) due to a complaint. Apparently, some of the code in the RNG was generating warnings and a user considered this enough of an annoyance to report it as a bug. The code in question read unallocated memory for seeding purposes, so the debugger reported it as an uninitiated variable. Unfortunately, it was not clear to the Debian package maintainers what that code was supposed to do, so they created a Debian-specific patch that removed the two lines in question. The result is that Debian *Linux* and all Debian-derived distributions had the bug for a while, meaning that a large number of systems are affected.

This affects many cryptographic protocols, in particular key generation and the Diffie-Hellmann key exchange protocol. The problem is that even though Debian has now been fixed and many systems have probably been updated, we may have to live with the bad keys for a long time. Luckily, users of *Firefox* are not susceptible (even if the web servers it connects to are) as it uses its own crypto library. This vulnerability also highlights how insecure code may be socially engineered into open or closed source software.

MD5 and CA Certs

The most eagerly anticipated event of the conference was the demonstration of an attack against a Certificate Authority (CA). The organizers didn't announce the title or content of this talk until a few hours before the event, for fear of a court injunction or similar action preventing it from going ahead.

We have known for many years now that the MD5 cryptographic hash function is broken and other hash functions are under attack (see *VB*, October 2004, p.13). There have been demonstrations of engineered MD5 hash collisions in the past. However, many people still continue to use MD5 in security applications – and, specifically, they are still being used in the generation of cryptographic certificates for websites and code.

Using the known prefix attack against MD5 and deriving the way that *Equifax*'s *RapidSSL* service generates website certs, the researchers were able to create an MD5 collision for a CA certificate using the site cert generated for them by *RapidSSL*. This was a clever piece of engineering as the first fields of a cert are generated by the CA, so they needed a way of predicting these, while the rest of the certificate was under their direct control. The prefix attack allowed them to create a CA cert document for which the signature of the real CA still held true.

With this rogue CA certificate, the researchers could create any sort of site or code certificate they wanted and it would look like the rogue certificate authority was certified by *Equifax*. They hobbled the generated CA cert by backdating its expiry date to 2004 (the year of the first publication of hash collisions) so that any certificates it generated would not be accepted in practice. While experimenting with this, they found that most browsers do not honour the certificate revocation system, the notable exception being *IE* in *Vista*.

The attack is only possible because of the weakness of MD5, and the moral of the story is that we need to stop using MD5 in security applications as soon as possible. The researchers recommended SHA-1, although it too is beleaguered, and so it might be prudent to move beyond SHA-1 and implement a certificate management scheme.

We had a lively debate about the significance of this presentation. I think it is a significant proof of concept, and even though the researchers picked on *RapidSSL*, there is a good chance that someone will find a way to abuse a different CA as well. It is always important to have a proof of concept to kick people into action. However, we've known for some time that MD5 is broken. We also know that people don't pay enough attention or don't understand the certificate system when browsing. Given the DNS and RNG problems, authentication in the Internet is pretty broken at this point anyway. So, it may not make much of a difference. What worried me was that when I met the speakers briefly, my impression was that there is more in the pipeline – they did not reveal everything they knew. It will pay to be vigilant.

The Cisco IOS

Felix Lindner (FX) talked about attacks against the *Cisco IOS*. Basically, if motivated enough, instead of

concentrating on servers, an attacker can take control of the infrastructure instead. Since *Cisco* is, by a large margin, the market's biggest supplier of networking equipment, it makes sense to concentrate on *Cisco*'s *IOS* operating system. Having said that, the attacker would need to be pretty motivated and it would have to be a part of a targeted attack. The *IOS* comes in many versions compiled in different ways by different engineers. In fact, OS diversity is probably *Cisco*'s best defence, as the OS itself is just as susceptible as any piece of software. It would also be a very effective attack and would give the attacker control over the infrastructure. To make matters worse, FX explained how difficult it is to perform post-mortem forensic analysis on *IOS*: if an error occurs, the equipment reboots itself, removing the evidence of the attack.

To counteract this, FX has been working on tools to log and diagnose the state of the IOS. He also suggests one simple solution: restrict connections to the infrastructure machines, as there is no reason why a user PC needs to access the router directly.

MALWARE

There were a few talks on malware, but little of any real significance. All I really learned from a talk on SWF malware is that Actionscript 3 will make *Flash* much more interesting, but it is still rarely used. Future versions of *Flash* may include P2P and UDP libraries, making it something worth keeping an eye on. Oddly enough, the 'Curse of Silence' attack against some S60 phones was shown as a part of the 'Security Nightmare' session. Apart from that, most of the talk was tongue-in-cheek speculation.

Storm

While the Storm worm seems to be subsiding a little, it is still a fascinating beast. There was a talk about how to take over parts of the Storm net. This relies heavily on understanding how Overnet works and how it uses hashes. Basically, a Storm bot uses the Overnet to locate the C&C and then communicates directly with it. It turns out that one can poison the hash system and point the bots to one's own system. One needs to know how to pretend to be the C&C to do this, and the authors gathered their knowledge by reverse engineering the bots and looking at the traffic. They could have 'cleaned' all the bots in the network, but refrained from doing so. It probably would be illegal in most countries – and dangerous in any case.

OTHER TALKS

A new technology is coming to mobile phones: Near Field Communications (NFC). This is a sort of phone-enabled RFID and is being pushed by telcos but appears only to be available so far in one phone. The suggested uses for the technology include: fare-collection, physical access control, electronic tickets and social networking. To be used in such situations the technology would require really good security, but it looks like it is based on the rather broken Mifare (TM) system with the possibility of using some extra phone CPU power. The work of VU Amsterdam has shown that all sorts of security problems can be caused by bad RF security implementations.

Fabio Yamaguchi talked about vulnerabilities in the TCP protocol that can lead to DoS attacks. His argument is that the assumptions of the attack model made when TCP was designed are different from the current reality. They were more concerned with nodes and network connections being unavailable due to bombings or sabotage than an attack from inside. Apart from existing attacks that we have already seen, Yamaguchi talked about abusing the congestion flow control system to overload the Layer 5 servers or clients.

Tillmann Werner showed off a system that is able to extract signatures directly from the trace data of a honeypot. He had a few neat tricks to make it work smoothly. The system is called nebula (http://nebula.mwcollect.org/), and it refines the signatures as more similar attacks are caught.

CONCLUSIONS

As usual, the CCC Congress was very intense and informative. It is an opportunity to exchange ideas and discover new ones in an open environment. With three parallel tracks, workshops and other events on the side and parties at night, there is not much time to track down and talk to people one knows or wishes to get to know. The fact that nearly all of the talks are recorded relieves the pressure of squeezing oneself into the over-full talks to see them live, but it nearly obliges one to review the over 150 hours of video that was created to see what one missed. So, after surviving the four days, I usually spend weeks going over missed presentations while trying to get rid of whatever germs I picked up at the event.

The MD5 event was intended to be the highlight of this year's Congress, but on its own it failed to impress. However, taken together with the DNS, Debian RNG and some of the other problems that were discussed, a very bleak picture is being painted. It is emerging that the foundations of the Internet are shakier than we thought and the application layers are being developed with little thought for security. My thoughts when leaving the Congress were that 2009 is going to be a very interesting year – but not in a good way.

COMPARATIVE REVIEW – PROLOGUE

VB RAP TESTING

John Hawes

This month sees the introduction of a new testing format to *VB*'s bi-monthly VB100 comparative review – one of the biggest changes to the format of the reviews since the inception of the VB100 certification scheme over ten years ago.

The introduction of the new test is the first in a series of planned expansions and improvements to the review data provided by *Virus Bulletin* – part of a major push to keep our readers better informed of the capabilities of the ever-expanding range of security software on the market.

The development of the new scheme over the past few months has been a lengthy process, with several trial runs and much consultation. A preliminary set of proposals and trial results were first discussed with *VB*'s board of advisors and other trusted experts last summer, and a second trial round – the methodology adjusted based on initial feedback and with participating products anonymized – was presented to the industry at the *VB* conference in Ottawa last October. Having taken on board further advice and suggestions and further honed the setup, this month sees the test's debut in a comparative review (see page 17).

The new test, which we have called 'RAP' ('Reactive and Proactive') testing, has a fairly simple design.

REACTIVE

Once the product submission deadline for a comparative review has been set, we compile a collection of malware samples first seen in each of the three weeks prior to the deadline date. These are referred to as 'week -3', 'week -2' and 'week -1'. These test sets form the reactive part of the test, measuring how well product developers and labs have been able to keep up with the steady, and steadily growing, flood of new malware emerging every day across the world. Most of the samples included in these sets are from the daily collections shared between labs and other trusted organizations. They are generally considered to be of high priority, and thus most well-connected malware labs should have access to the samples at the same time as we see them, if not earlier. Establishing whether they can cope with processing and, if needed, adding detection for them is the main aim of this part of the test.

Prioritization is also a major issue here, and some labs may – quite rightly – see it as more important to ensure full detection of particularly prevalent or dangerous items, rather than obscure targeted trojans that are unlikely to reappear. To help cover this angle, we plan to do some prioritization of our own, aligning our sample selection processes with the prevalence data we gather from a range of sources – the aim being to include the most significant items. This is by no means a simple task – prevalence data comes in a variety of forms, many of which are proving increasingly difficult to match to specific items as family and variant group names become increasingly vague and generic. Incoming data with more detail, including specific file identifiers, would be of great help here, and we continue to seek more and better prevalence data to add to our incoming feeds.

Our second trial test included some comparisons between the detection rates achieved when scanning the full incoming feed and those achieved when scanning just those items adjudged to be particularly prevalent. Some very interesting results were obtained. However, part of the reason for filtering incoming samples by prevalence is to reduce the incoming feed to a manageable level which can be checked and validated in the short time available, it would therefore not be appropriate to include such additional data in a full comparative review.

PROACTIVE

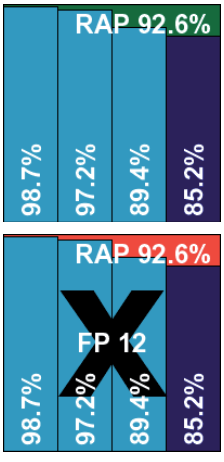
The second prong to this new test is the proactive angle. In addition to the three test sets compiled prior to the product submission deadline, a fourth set of samples is put together in the week following product submission ('week +1'). This set ought to consist mostly of samples that have not been seen by labs at the time of product submission, and thus will not be specifically catered for by targeted detection signatures. The purpose of this test set is to gauge products' ability to detect new and unknown samples proactively, using heuristic and generic techniques. Comparing the 'week +1' results to the results of the previous three weeks will provide insight into the extent to which vendors rely on proactive preparedness as opposed to rapid response.

This is quite a significant step for *VB*'s comparative testing, which has in the past set strict test set deadlines – for both malicious and clean items – a few days in advance of the product submission deadline, giving all participants time to ensure their products fully cover the samples in our sets. It also means that full testing cannot begin until a week after the product submission deadline. In the past, the products being tested have been taken in around a month prior to publication of the review, with testing and result processing proceeding throughout the month. As this is already a rather tight schedule – particularly with the growing number of products taking part in recent years – it may be necessary to set the deadlines slightly earlier, but we will endeavour to keep this schedule adjustment to a minimum, to ensure our results are as up to date as possible when published.

The adjustment in the timeline of the test will also put considerable pressure on our malware validation process,

which we endeavour to keep as strict as possible given the tight deadlines. We are hard at work attempting to automate the validation process as far as possible to get as many samples processed and included in the test sets as we can.

RESULTS



Astute readers will doubtless have an idea of the likely output of this new test regime. Our prediction from the outset has been that most products will show a slight decline in performance over the three reactive weeks, with detection strongest over the collection of samples seen longest ago ('week -3'), and a sharper downward step in detection for the proactive week ('week +1'). This pattern is expected to be especially pronounced for those products whose labs concentrate on fast reaction times over heuristics. In the trials this pattern was followed fairly well at a general level, but at an individual product level there were numerous surprises and anomalies, one particularly interesting trend being a poor showing by many products on the 'week -3' set compared to the 'week -1' set.

The test results will be represented graphically, as shown above. The three pale blue bars represent (from the left) weeks -3, -2 and -1, while the dark blue bar represents week +1. An overall 'RAP score' is also presented on the graph, which represents the average detection over the four weeks. In cases where products have generated false positives in our tests the background of the graph will be coloured red and a large cross, together with 'FP=' will act as a warning to the user, showing the number of false positives generated.

Such a wide variety of factors affect the test – from sample selection and classification to national holidays – that such oddities are bound to occur, but any true anomalies should be evened out over the course of time, with repeated tests leaving behind genuine quirks. One of the most interesting aspects of this test will be the picture that builds up over time, as strings of results are put together to show long-term trends and patterns for specific products.

As work on automating various aspects of our lab processes continues, in between busy comparative months we hope to continue to build test sets and measure performance. However, it is likely that this data – generally based on untrusted, unvalidated samples – may only be made available to labs themselves, and probably in partially anonymized format. The extra data gathered in this way may provide even more fine-grained insight when viewed over the long term, and we hope to present periodic analysis of such data in these pages once enough has accumulated.

In order to build up this timeline of results, we must of course start somewhere. So, this month we publish the first set of figures. From the initial idea of RAP testing to this first release there have been numerous tweaks to the format, but the only real test of its viability is its implementation in the setting of a real, busy VB100 comparative review, using genuine, often complex and intractable products. Doubtless this first full outing will highlight a range of issues not previously anticipated, and the format will need a few more tweaks and adjustments. As with the VB100 results themselves, readers should refrain from putting too much faith in a single set of RAP results, but be patient and wait for true patterns to emerge over time.

VB100

The results of the RAP tests will form part of the additional data provided in our comparative reviews, alongside other extras such as the detection results of our zoo test sets, our speed and on-access overhead measurements, and my (somewhat subjective) overviews of design and usability.

As such, the introduction of the RAP test does not affect the basic tenets of the VB100 certification programme: the requirement for products to detect the full WildList, both on access and on demand, without false positives. These central certification standards remain unchanged, although we expect to revamp the fine print of the certification procedures in the near future.

In particular, we feel that the definition of 'on access' has become somewhat over-specific, with more and more protection software including a wealth of behavioural and HIPS technologies which require full execution of malicious code before they step in. Whether such mechanisms, with their wide sliding scales of detection, warning and blocking, can be included neatly in the VB100 rules without compromising the black-and-white nature of the programme, is something that requires a little more thought and investigation.

As for fully testing the complete range of offerings in the latest generation of desktop suites, including online black- and whitelisting resources, integrated firewalls, web and mail filters and much more besides, it seems likely that this will require a radically new and entirely different approach – something we are working hard on developing. We will, however, continue to run the VB100 certification scheme (with appropriate modifications where necessary) for as long as our readers find it informative and useful.

VB welcomes readers' feedback on the proposed test methodology. Please direct comments and enquiries to john.hawes@virusbtn.com.

COMPARATIVE REVIEW

RED HAT ENTERPRISE LINUX 5.2

John Hawes

Once again our annual visit to the *Linux* platform has rolled around. The relatively small number of participants in this month's test is in part due to the limited number of vendors that provide support for the platform, but has been further reduced by the unexpected withdrawal from the test of several of our regulars. Reasons given for sitting this one out included ongoing engine update work, difficulties coping with a deadline close to the new year, and a simple lack of organization in preparing a product for submission. However, past experience has taught us that any time saving introduced by having a diminished field of competition can be more than outweighed by the additional complexities introduced by the *Linux* platform. Based on my previous visits to the platform and the acidic comments of past reviewers, we expected problems with recalcitrant, opaque and poorly documented products as well as unexpected dependencies and incompatibilities.

More significantly, this month's comparative review sees the introduction of a new set of test results to our battery of additional data. Our RAP (Reactive and Proactive) testing setup, developed over the last few months and presented to the industry at last year's *VB* conference in Ottawa, makes its debut in these pages, and should provide some interesting insights into the products' performance. Its full value will, we hope, emerge in the long term, as further refinements are made and long-term trends are analysed – the new system and the intentions behind it are discussed in more detail on page 15.

PLATFORM AND TEST SETS

The last VB100 on *Red Hat Linux* was in 2006 (see *VB*, April 2006, p.13), until which time it had dominated the *Linux* slot in the comparative schedule for several years. Past reviewers tended to focus on the freely available and hugely popular *Red Hat* 9, the last version of which was released before the split between *Red Hat* and *Fedora*. In the intervening years, while we have turned our attention to commercial arch-rival *Novell/SuSE* (see *VB*, April 2007, p.11) and the more freely available *Ubuntu* (see *VB*, June 2008, p.16), *Red Hat* has continued down commercial lines, producing a line of business-focused distributions backed up by broad support offerings. These continue to hold a strong position in the blossoming market for open-source operating systems in business, while hobbyists and home-users alike have formed great attachments to the *Fedora* variant. The latest iteration of the commercial product, *RHEL 5.2*, was released in mid-2008, and while a further update to the 'Tikanga' line, version 5.3, was due for release halfway through this month's test, it seemed appropriate to stick with the edition most likely to be in use in *Red Hat*-based enterprises.

Installation and setup of the test systems was relatively straightforward at first. Following the simple and unfussy installer interface through and selecting the defaults as far as possible proved a simple and trouble-free task, and the GUI presented once up and running was equally free from excess glitter. The look and feel seemed fairly plain and clunky next to the beauty of the latest generation of desktops, but as a serious and sensible desktop for a server admin, it seems fit for purpose. Numerous graphical sysadmin tools are provided for those not keen on getting their hands dirty meddling with configuration files, and after some time finding our way around the anomalies and eccentricities of the system layout, things were mostly as we wanted them.

A few initial annoyances presented themselves, not least of which was the complete absence of NTFS support in the standard installation. As the test machines carry an NTFS partition hosting a number of useful lab items, some extra installation and configuration work was required – but nothing too taxing. Configuring the *Samba* daemon to make a storage area on each test system visible to *Windows* was also a fairly simple task. A separate system was positioned alongside the standard test machines for the purposes of the on-access tests. The system was running a basic *Windows XP Pro SP3* setup with the samba share from each of the test systems mounted. This would represent our client machine, accessing network resources and, hopefully, being protected from anything malicious which might be lurking on shared storage. Tests from here would include speed tests, which would be run separately with minimal network activity, to reduce the impact of additional traffic on the speed measurements.

The final stage of preparing the systems was to provide the open-source file-hooking module *dazuko*, which we knew from experience would be required by many products for their on-access scanning. As in previous *Linux* comparatives, getting this up and running proved less straightforward than was suggested by the accompanying documentation. The default kernel included with the operating system had a built-in module which turned out to be incompatible with *dazuko*. As a result, the kernel had to be recompiled without the module – which was by no means an arduous task, but certainly a time-consuming one. For the purposes of the test we simply made the alternative kernel and module available from the start, but this extra labour would have to be counted against those products using the system as far as ease of setup was concerned.

On-demand detection	WildList viruses		Worms & bots		Polymorphic viruses		Trojans		Clean sets	
	Missed	%	Missed	%	Missed	%	Missed	%	FP	Susp.
Alwil avast!	0	100.00%	1	99.95%	20	97.02%	64	98.20%	0	0
Avira AntiVir/Linux	0	100.00%	0	100.00%	0	100.00%	301	91.44%	0	0
ESET Security	0	100.00%	0	100.00%	0	100.00%	458	87.01%	0	0
Frisk F-PROT AntiVirus	0	100.00%	0	100.00%	174	96.08%	986	72.04%	1	0
F-Secure Linux Security	0	100.00%	1	99.95%	78	99.47%	458	86.99%	0	1
Kaspersky Anti-Virus	0	100.00%	0	100.00%	72	99.56%	464	86.83%	0	0
McAfee LinuxShield	0	100.00%	0	100.00%	0	100.00%	568	83.89%	1	0
Quick Heal for Linux	0	100.00%	48	97.43%	914	87.22%	717	79.67%	0	0
Sophos Anti-Virus	0	100.00%	2	99.97%	725	91.13%	562	84.06%	0	0
Symantec AntiVirus	0	100.00%	0	100.00%	0	100.00%	671	80.97%	0	0
VirusBuster SambaShield	0	100.00%	2	99.96%	757	81.43%	1369	61.17%	0	0

Next, the test systems were loaded with the test sets. The core detection set, based on the November 2008 WildList, had moved on considerably since the previous test, with large numbers of long-term residents finally evicted, including the bulk of the W32/Mytob, W32/Sdbot, W32/Rbot, W32/Stration (aka Warezov) and other worms which had dominated the list for several years.

Also falling off the list were the last of the W32/Virut polymorphic file-infecting viruses which had caused a considerable stir since their appearance last year. This left the list pretty devoid of genuine viruses and made up mostly of banking and online gaming password stealers. Many of the retired items which continue to show up in small numbers in our prevalence reports were moved temporarily across to our worms and bots test set. We hope, in future, to replace the worms and bots test set entirely for each new test, in the same manner as the trojan set – which, once again, was compiled from items seen in the few months prior to the test, and categorized into prevalent family groups.

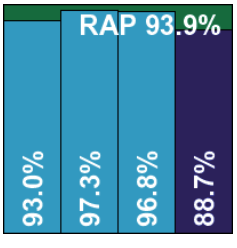
Still more recent items were put into the sets for the new RAP test. With the test deadline set for 7 January, the three 'reactive' sets were compiled from samples first seen in the last two weeks of 2008 and the first week of 2009, with the 'week +1' set compiled using samples seen in the week following product submission. Perhaps due to the change of year and various holidays upsetting the routines of both malware creators and external sample sources, the sets varied considerably over this period in both size and content type. After filtering, the final week's test set contained rather fewer samples than we had hoped, but still enough to give a reasonable reflection of detection abilities. Any anomalies caused by the makeup of the sets should be evened out over time – as with VB100 results, readers should not place too much importance on a single set of RAP results, but wait for true patterns to emerge as the tests are repeated over time.

Finally, the clean sets went through their usual tidying and expansion, with a fairly large selection of new samples added. New additions included the contents of a batch of cover CDs from technical magazines and a selection of packages broadly categorized as web-browsing and media manipulation tools. Although the expansion of this set was limited due to the amount of time devoted to preparing the other sets (and by a well-earned December break), the new additions seemed likely to challenge products coming up against our strict no-false-positives rule.

With all of the test collections in place, it was time to start feeling our way, with great caution, around the selection of products submitted for review.

Alwil avast! 3.1.5

ItW	100.00%	**Polymorphic**	97.02%
ItW (o/a)	100.00%	**Trojans**	98.20%
Worms & bots	99.95%	**False positives**	0



Alwil's *avast!* product for *Linux* arrived as a trio of RPM packages, one of which included an attempt to adjust the crontab scheduler to automate updates. This seemed to be taking some time, so was aborted, but some initial tinkering with the product found it still to be inactive. The instructions provided by the developers revealed that this was not the result of our impatience, but rather the requirement for a licence file, provided along with the submission but which needed to be copied

On-access detection	WildList viruses		Worms & bots		Polymorphic viruses		Trojans		Clean sets	
	Missed	%	Missed	%	Missed	%	Missed	%	FP	Susp.
Alwil avast!	0	100.00%	1	99.95%	20	97.02%	64	98.20%	0	0
Avira AntiVir/Linux	0	100.00%	0	100.00%	0	100.00%	301	91.44%	0	0
ESET Security	0	100.00%	0	100.00%	0	100.00%	468	86.72%	0	0
Frisk F-PROT AntiVirus	0	100.00%	0	100.00%	174	96.08%	986	72.04%	1	0
F-Secure Linux Security	0	100.00%	1	99.95%	428	99.56%	537	84.77%	0	1
Kaspersky Anti-Virus	0	100.00%	0	100.00%	500	96.97%	591	83.24%	0	0
McAfee LinuxShield	0	100.00%	0	100.00%	0	100.00%	568	83.89%	1	0
Quick Heal for Linux	0	100.00%	48	97.43%	914	87.22%	1466	58.42%	0	0
Sophos Anti-Virus	0	100.00%	0	100.00%	725	91.13%	562	84.06%	0	0
Symantec AntiVirus	0	100.00%	0	100.00%	0	100.00%	754	78.62%	0	0
VirusBuster SambaShield	0	100.00%	2	99.96%	757	81.43%	1369	61.17%	0	0

manually into the appropriate location, as indicated by a configuration file.

With these initial tasks complete, running the product proved straightforward, with the syntax of the command-line scanner a little esoteric but clearly laid out in the accompanying instructions. On-access scanning was similarly straightforward to administer, via standard and lucid configuration files, and everything ran pretty smoothly. Scanning speeds were quite excellent, both on access and on demand, and detection rates at their usual exemplary level.

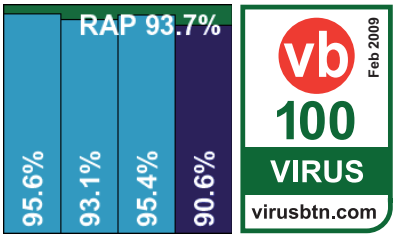
The RAP scores showed a slight dip in the earliest week – which, logically, one would expect to have the best coverage, but the coinciding holidays in many territories may have affected the throughput of labs in this period. More in tune with predictions, a second dip was observed in the 'week +1' set compiled after update freezing, but detection remained pretty solid over these likely unseen samples.

Getting back to the VB100 certification requirements, with no trouble at all handling the diminished WildList set and not a whisper of a false positive, *Alwil* takes the first VB100 award of 2009 with considerable style.

Avira AntiVir/Linux 2.1.12-101

ItW	100.00%	**Polymorphic**	100.00%
ItW (o/a)	100.00%	**Trojans**	91.44%
Worms & bots	100.00%	**False positives**	0

Avira's product came in the form of a single .tgz file. This raised some concerns initially, but on extracting, the file proved to contain an install script which performed all the necessary setup steps clearly and simply, with a series of simple questions allowing basic user configuration. With the locations of the licence file and *dazuko* module provided as part of the setup process (the *dazuko* module is developed and maintained by *Avira*), things were up and running in no time, and after perusing another well-documented, but again slightly eccentric set of command-line qualifiers, testing zipped along nicely.



Speed was highly impressive on demand, but on-access scanning seemed a little sluggish by comparison, and little difference was noted in speeds when archive scanning was activated (not the default setting). Indeed, it seemed the same kind of analysis was being performed – it took a considerable time to get through the control archive test set (consisting of the EICAR test file embedded at different depths inside a variety of archive formats) on access, without any detection being made, and not noticeably longer when full scanning was activated and access to the test files was correctly denied.

Detection rates were again superb, with over 90% across the board in all RAP sets including the 'week +1' set.

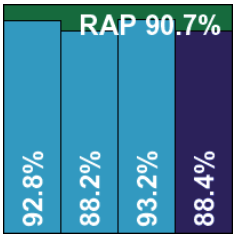
No detections were missed in the WildList set and no false alarms were generated in the clean sets, thus *Avira* starts 2009 with a VB100 award and great respect.

ESET Security for Linux 3.0.10

ItW	100.00%	**Polymorphic**	100.00%
ItW (o/a)	100.00%	**Trojans**	87.01%
Worms & bots	100.00%	**False positives**	0

On-demand throughput (MB/s)	Archive files				Binaries and system files				Media and documents				Other file types				Linux files			
	Default settings		All files		Default settings		All files		Default settings		All files		Default settings		All files		Default settings		All files	
	Time (s)	Thr. put (MB/s)	Time (s)	Thr. put (MB/s)	Time (s)	Thr. put (MB/s)	Time (s)	Thr. put (MB/s)	Time (s)	Thr. put (MB/s)	Time (s)	Thr. put (MB/s)	Time (s)	Thr. put (MB/s)	Time (s)	Thr. put (MB/s)	Time (s)	Thr. put (MB/s)	Time (s)	Thr. put (MB/s)
Alwil avast!	364	8.37	564	5.40	230	11.30	236	10.99	86	24.06	98	21.13	93	10.11	98	9.65	511	1.85	530	1.78
Avira AntiVir/Linux	43	70.69	351	8.67	159	16.38	164	15.81	81	25.39	90	22.83	96	9.80	112	8.41	528	1.78	1623	0.58
ESET Security	631	4.82	631	4.82	414	6.27	414	6.27	71	29.16	71	29.16	92	10.28	92	10.28	1056	0.89	1056	0.89
Frisk F-PROT AntiVirus	308	9.87	309	9.83	400	6.49	436	5.96	69	29.98	100	20.57	74	12.68	109	8.67	393	2.40	806	1.17
F-Secure Linux Security	4577	0.66	4577	0.66	1110	2.34	1110	2.34	372	5.54	372	5.54	438	2.15	438	2.15	5610	0.17	5610	0.17
Kaspersky Anti-Virus	2471	1.23	2471	1.23	653	3.98	653	3.98	159	12.96	159	12.96	196	4.82	196	4.82	2561	0.37	2561	0.37
McAfee LinuxShield	718	4.24	718	4.24	435	5.97	435	5.97	84	24.57	84	24.57	105	8.97	105	8.97	1234	0.76	1234	0.76
Quick Heal for Linux	519	5.87	519	5.87	152	17.10	152	17.10	89	23.26	89	23.26	123	7.69	123	7.69	1452	0.65	1452	0.65
Sophos Anti-Virus	79	38.37	1302	2.34	314	8.28	337	7.71	65	31.59	103	20.12	31	30.24	134	7.06	283	3.33	2010	0.47
Symantec AntiVirus	158	19.28	NA	NA	213	12.21	213	12.21	128	16.11	128	16.11	97	9.67	97	9.67	802	1.17	NA	NA
VirusBuster SambaShield	382	7.96	643	4.73	243	10.68	246	10.56	255	8.09	181	11.42	128	7.35	148	6.36	1275	0.74	1638	0.58

ESET's long-standing dominance in VB100 testing has been challenged of late, both in terms of speed and detection rates, by some strong up-and-comers, with two of its most pressing rivals having already appeared in this month's review. Installation of the product was in the form of a single, straightforward RPM package, with control of the program via a centralized configuration file and thorough, well-documented options to the main binary. The default settings were pretty thorough, covering all file types and a wide set of archive types, and speeds in both modes were as excellent as experience has led us to expect.

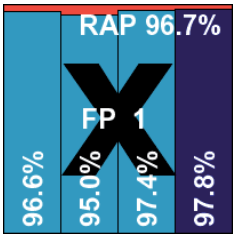


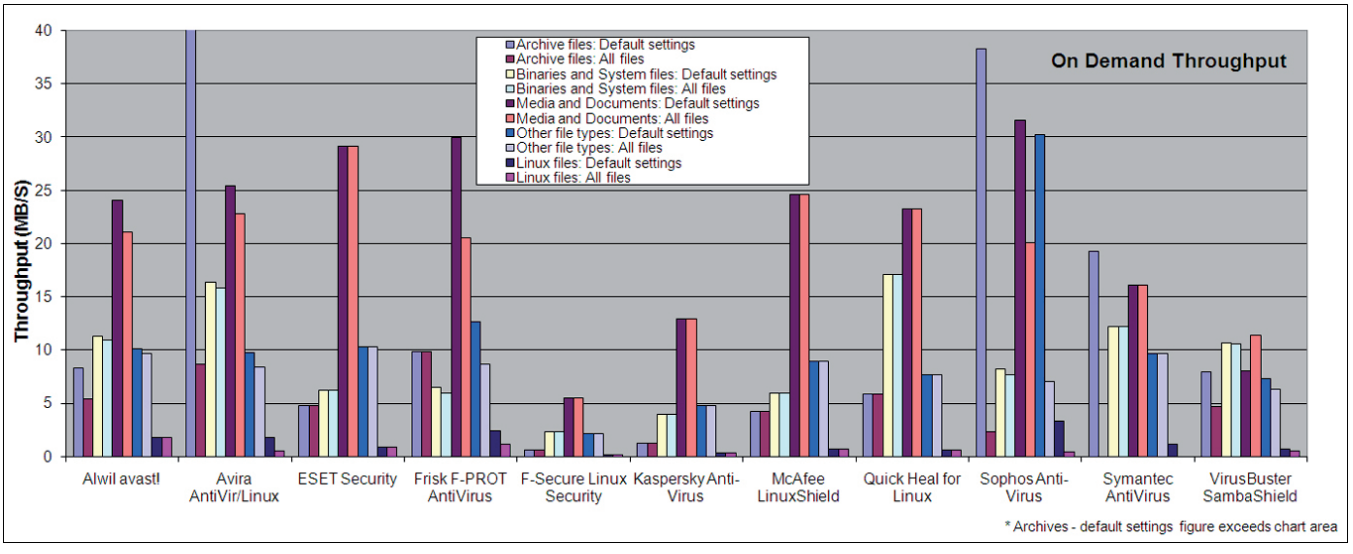
Detection rates were similarly strong – perhaps a fraction behind the excellent performers seen so far in the RAP tests, but close enough to put down to sample selection anomalies at this early stage. As usual for *ESET*, false positives were absent despite the product's strong heuristics, and the set of WildList samples presented no problems, thus *ESET* continues its excellent run of success with another VB100 award and a performance worthy of respect.

Frisk F-PROT AntiVirus for Linux 6.2.1.4252

ItW	100.00%	**Polymorphic**	96.08%
ItW (o/a)	100.00%	**Trojans**	72.04%
Worms & bots	100.00%	**False positives**	1

Frisk's *F-Prot* has shown itself in recent *Windows* comparatives to be the champion of pared-down, no-fuss protection, and here, once again, is an extremely basic piece of anti-malware kit – and none the worse for it. Installation consisted of little more than extracting an archive containing the required files, which could thus be located wherever the admin desires with relative ease. A simple install script is also provided to set up default paths to binaries and man pages. An initial problem was encountered when the submitted product turned out to be a simple workstation version with





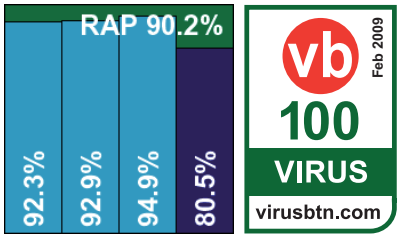
no on-access component. However, this issue was soon circumvented by grabbing the server version – available as a free trial download from the vendor's website – and snaffling the required on-access components for interaction with the *dazuko* module, which proved more than adequate to provide the full level of protection.

As expected, given the pared-down nature of the product, speeds and overheads were exemplary. Detection rates were decent across the three reactive RAP sets, and in the newest set quite remarkable, producing somewhat eye-opening results. Further investigation showed that a fair proportion of the detections recorded when parsing the results were in fact rather vague – many of them being labelled simply 'security risk' or even 'possible security risk'. Such detection labels would not be counted as false positives in the full test, so it was somewhat difficult to decide whether they should count as full detections in this case, but with time pressing and much of the processing of results already completed, we had no choice but to leave them in.

Moving on to the VB100 certification requirements, the WildList was once again covered without issues, but in the clean sets a single file from this month's addition of web browsers and associated tools was erroneously flagged as a backdoor. While there is potentially some scope for the item in question – a cookie management tool – to be abused, the alert was judged sufficient to deny *Frisk* a VB100 award this month despite the product's otherwise solid detection rates.

F-Secure Linux Security 7.02.73807

ItW	100.00%	**Polymorphic**	99.47%
ItW (o/a)	100.00%	**Trojans**	86.99%
Worms & bots	99.95%	**False positives**	0



F-Secure's product presented a much more professional aspect, with a .tgz file containing the required components alongside a thorough install script which leads the installer through all the required steps to get the product set up. This includes its own copy of the *dazuko* module – something which none of the other products so far have provided (despite requiring it for their on-access protection). It also comes with an attractive web-based interface, which plants its own desktop icon and provides configuration for much of the product. Along with numerous components in the init directory, a range of additional utilities are provided for the configuration and operation of the product, which provides a full protection suite, including firewall, alongside the standard anti-malware protection.

As expected, once the vagaries of the command-line interface had been decoded, helped along by clear documentation, detection rates were pretty solid, although less than perfect on some of the new families of polymorphic viruses. Scanning speeds were somewhat leisurely – which can partly be explained by the multiple engines in use by the product, which appear to contribute strongly to the depth of detection. Even using the default on-access settings, which ignored most archive types entirely, speeds were notably slower over the clean speed sets than for some of the other products. However, detection rates in the new RAP tests were once again excellent, with a notable dip in the 'week +1' set containing samples unlikely to have been seen by labs. For *Windows* users, *F-Secure* has

File access lag time (s/MB)	Archive files				Binaries and system files				Media and documents				Other file types				Linux files			
	Default settings		All files		Default settings		All files		Default settings		All files		Default settings		All files		Default settings		All files	
	Time (s)	Lag (s/MB)	Time (s)	Lag (s/MB)	Time (s)	Lag (s/MB)	Time (s)	Lag (s/MB)	Time (s)	Lag (s/MB)	Time (s)	Lag (s/MB)	Time (s)	Lag (s/MB)	Time (s)	Lag (s/MB)	Time (s)	Lag (s/MB)	Time (s)	Lag (s/MB)
Alwil avast!	629	0.20	629	0.20	284	0.09	284	0.09	242	0.08	242	0.08	176	0.12	176	0.12	1978	2.03	1978	2.03
Avira AntiVir/Linux	1973	0.65	1976	0.65	221	0.07	224	0.07	165	0.04	187	0.05	272	0.22	272	0.22	23430	24.80	2564	2.65
ESET Security	657	0.21	657	0.21	489	0.17	489	0.17	166	0.04	166	0.04	171	0.11	171	0.11	2019	2.07	2019	2.07
Frisk F-PROT AntiVirus	290	0.09	292	0.09	424	0.14	430	0.15	142	0.03	147	0.03	127	0.06	131	0.07	1469	1.49	2231	2.30
F-Secure Linux Security	234	0.07	4569	1.50	950	0.35	1147	0.42	355	0.13	415	0.16	408	0.36	462	0.42	1530	1.55	17443	18.44
Kaspersky Anti-Virus	74	0.02	2169	0.71	652	0.23	678	0.24	232	0.07	237	0.07	239	0.18	245	0.19	3146	3.27	3953	4.13
McAfee LinuxShield	592	0.19	592	0.19	609	0.22	609	0.22	235	0.07	235	0.07	228	0.17	228	0.17	2647	2.74	2647	2.74
Quick Heal for Linux	32	0.01	NA	NA	194	0.06	194	0.06	165	0.04	165	0.04	179	0.12	179	0.12	2480	2.56	2480	2.56
Sophos Anti-Virus	91	0.03	761	0.25	357	0.12	389	0.13	168	0.04	170	0.04	165	0.11	173	0.11	2486	2.57	2857	2.96
Symantec AntiVirus	163	0.05	NA	NA	253	0.08	253	0.08	196	0.05	196	0.05	156	0.10	156	0.10	1842	1.89	NA	NA
VirusBuster SambaShield	56	0.02	NA	NA	312	0.10	312	0.10	237	0.07	237	0.07	204	0.15	204	0.15	2590	2.68	NA	NA

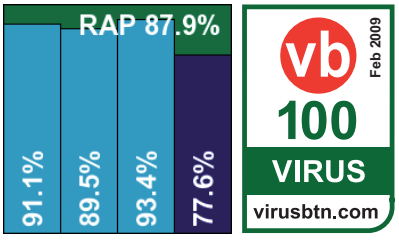
made much of its additional 'Deepguard' protection with additional cloud-based black- and whitelists (which we have yet to be able to properly test under the requirements of the VB100); whether this layer is available for *Linux* users was not made clear.

Overall, the product's performance was nothing to be sniffed at, with the WildList test set covered without a glitch, and the cleanliness of the clean sets was only called into question by a 'potentially unwanted' alert on the same file as was described as a backdoor by the *Frisk* product. As this is allowable under the VB100 rules, *F-Secure* earns a VB100 award, and extra praise for its solid and lucid design and usability.

Kaspersky Anti-Virus for Linux File Servers 5.7-26

ItW	100.00%	**Polymorphic**	99.56%
ItW (o/a)	100.00%	**Trojans**	86.83%
Worms & bots	100.00%	**False positives**	0

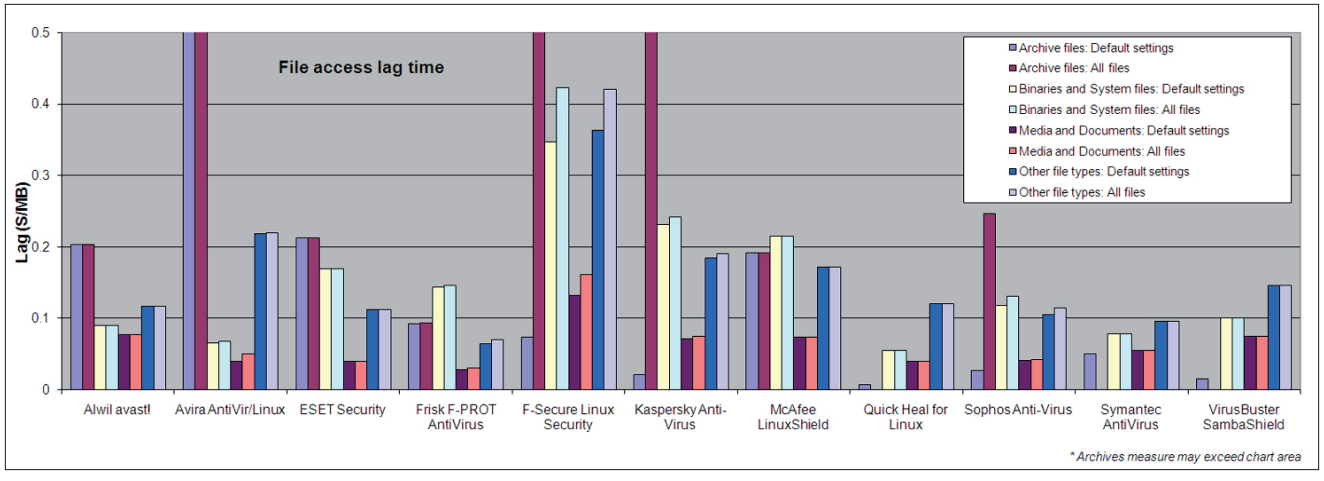
Kaspersky's *Linux* range has in the past eschewed the popular *dazuko* path in favour of the 'Samba vfs object' method, functioning only on file systems shared via *Samba*. In previous tests *Kaspersky* has proved to be one of the few vendors to utilize the technology to its full efficiency. This time, however, the vendor seems to have moved on to its own in-house technology, implementing full on-access detection without the need for any external software. Installation takes the form of an RPM, with a perl script to be run post installation to perform the necessary setup steps.



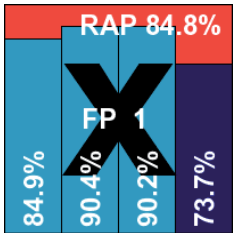
Operation of the product was less well streamlined, with lengthy commands needing to be issued from the command line and somewhat unpredictable syntax. Scanning speeds were not the fastest, but detection rates were at their usual solid level. The product showed another splendid performance in the RAP tests with, as was predicted for all products, a slight decline in the 'week +1' set. The diminutive WildList yet again presented no difficulties, and false positives were absent, thus earning *Kaspersky Lab* another VB100 award.

McAfee LinuxShield 1.5.1

ItW	100.00%	**Polymorphic**	100.00%
ItW (o/a)	100.00%	**Trojans**	83.89%
Worms & bots	100.00%	**False positives**	1



McAfee's *Linux* product has, in previous comparative reviews, reflected the company's reputation for professionalism and seriousness. Here, that impression was bolstered, with a broad collection of PDF-format documentation needing to be read before the required installation order of the RPM packages could be ascertained. This done, and after a standard array of installation questions, the product was quickly up and running, with administration performed via a fairly clear and thorough web interface.



A few changes have clearly taken place since my previous encounter with the product however, and a chink in the company's armour emerged when it became clear that these changes had yet to filter through to the online documentation, knowledgebase and even the staff submitting the product. The update method – admittedly not the standard online method, but one certain to be preferred by many *Linux* systems administrators who may be running their servers behind all kinds of protective barriers – had been adjusted with a recent iteration of the product, rendering previous techniques ineffectual and online instructions inaccurate. Eventually, after much discussion with tech support personnel, the problem was diagnosed and the correct form of updates provided, albeit not quite the freshest possible from the submission date. A more accurate set of instructions was also provided, and testing continued.

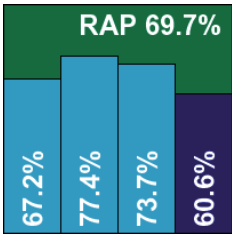
The running of on-demand scans required the use of the interface from which scan 'tasks' could be designed and run; these same tasks could also be kicked off from the command line, allowing for some scripting and the use of the standard cron scheduler. However, the lack of ability to configure the tasks from the bare console, even to the extent of providing a scan target, seemed a rather glaring omission which the diehard command-line-loving *Linux* administrator may find hard to forgive.

Once the scans were set up and run, scanning speeds were surprisingly good, overheads not too heavy, and detection rates in the standard test sets reached the expected level. In the RAP tests, scores were generally pretty good, with that telltale dip in the 'week +1' set demonstrating the superior performance of signature detections over heuristic and generic methods, but a worthy performance nevertheless. As far as certification requirements were concerned, nothing was missed in the WildList, but in the clean sets a single file – which has been included in the set since the summer of 2007 – was alerted on with a generic trojan identification, thus spoiling *McAfee*'s recent run of success and denying the vendor a VB100 on this occasion.

Quick Heal for Linux 10.00

ItW	100.00%	**Polymorphic**	87.22%
ItW (o/a)	100.00%	**Trojans**	79.67%
Worms & bots	97.43%	**False positives**	0

Quick Heal's product is another *dazuko*-based setup, with a nice, simple installer inside a .tgz file which, for once, utilizes colour to improve clarity and ease of use. With the setup completed quickly and easily, a proper desktop interface was another pleasant surprise, but although easy on the eye it provided little in the way of in-depth configuration. Some was available in more traditional configuration files, but even here some functions, such as enabling of archive scanning on access, seemed impossible.



		ACE	CAB	JAR	LZH	RAR	TGZ	ZIP	ZIP-SFX	EXT*
Alwil avast!	OD	X/√	X/√	√	X/√	X/√	√	√	X/√	√
	OA	√	√	√	√	√	√	√	√	√
Avira AntiVir/Linux	OD	X/2	X/√	X/√	X/√	X/√	X/√	X/√	X/√	√
	OA	X/2	X/√	X/√	X/√	X/√	X/√	X/√	X/√	√
ESET Security	OD	√	√	√	√	√	5	√	√	√
	OA	√	√	√	√	√	5	√	√	√
Frisk F-PROT	OD	X	5/√	5/√	√	5/√	5	5/√	5/√	√
	OA	1	5/√	5/√	X	5/√	2/5	5/√	5/√	√
F-Secure Linux Security	OD	√	6	6	6	6	3	6	6	√
	OA	X/√	X/√	X/√	X/√	X/√	X/5	X/√	X/√	√
Kaspersky Anti-Virus	OD	√	√	√	√	√	√	√	√	√
	OA	X/√	X/√	X/√	X/√	X/√	X/√	X/√	X/√	√
McAfee LinuxShield	OD	√	√	√	√	√	√	√	√	√
	OA	2	√	√	√	√	√	√	√	√
Quick Heal AntiVirus	OD	2	√	√	X	√	1	√	X	√
	OA	2	X	X	X	X	X	X	X	√
Sophos Antivirus	OD	X	X/5	X/5	X/5	X/5	X/5	X/5	X/5	X/√
	OA	X	X/√	X/√	X/√	X/√	X/8	X/√	X/8	√
Symantec AntiVirus	OD	X	X	3	3	3	1	3	3	√
	OA	X	X	3	3	3	1	3	3	√
VirusBuster Sambashield	OD	√	√	√	√	√	√	√	√	√
	OA	X	X	X	X	X	X	X	X	√

Key:

X - Archive not scanned

√ - Archives scanned to depth of 10 or more levels

*Executable file with randomly chosen extension

X/√ - Default settings/thorough settings

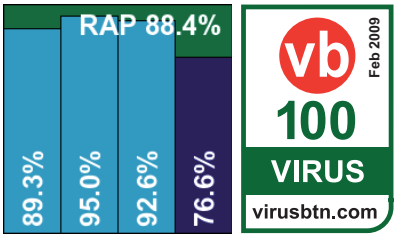
[1-9] - Archives scanned to limited depth

Of course, all of this helped with *Quick Heal*'s famous speediness, which was once again up there with the best. Although detection rates were a little behind the best scores recorded so far in this review, in most sets they were decent, and the WildList presented no problems. With no false positives either, *Quick Heal* reaches the required standard and earns another VB100 award.

Sophos Anti-Virus for Linux 6.4.5

ItW	100.00%	**Polymorphic**	91.13%
ItW (o/a)	100.00%	**Trojans**	84.06%
Worms & bots	99.97%	**False positives**	0

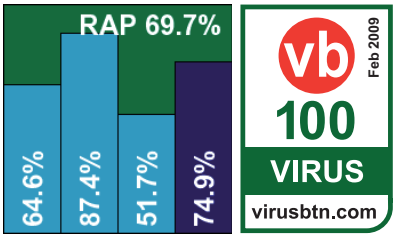
Another company with a solid reputation in the enterprise market, *Sophos* also ignores the availability of the *dazuko* module and goes for its own in-house technology to provide on-access scanning. The product installs simply and smoothly on this platform, which is a prime target as one of the leading *Linux* setups in use in enterprise. The absence of any recompilation, dependencies or other fiddly tasks counts strongly in the product's favour as far as initial installation goes. Post-install operation is also something of a breeze, with a well-documented and pleasantly usable product. Alongside the standard command-line operation and configuration files, another web interface is provided, which, again, is very well laid out and simple to use.



Scanning speeds were similarly pleasing, particularly with the default settings, and lag times were among the best on offer. Detection rates across all test sets left little to be desired, with a few misses in the polymorphic and trojan sets more than made up for by an excellent showing across the new RAP sets, although the dip in the 'week +1' set was perhaps a little more pronounced here than elsewhere. With no false positive issues and nothing missed in the WildList set, *Sophos* comfortably achieves a VB100 award.

Symantec AntiVirus for Linux 1.0.7.14

ItW	100.00%	**Polymorphic**	100.00%
ItW (o/a)	100.00%	**Trojans**	80.97%
Worms & bots	100.00%	**False positives**	0



Somewhere in the deeper circles lies a special hell, where testers who have devoted their lives to the more unforgiveable sins must forever wrestle fruitlessly with the *Symantec Linux* product. After two previous encounters with the product, and despite being given some insight into its intricacies by a gifted support engineer, it remains opaque and bizarre. The company's support forums are littered with desperate cries for help and simple requests for an explanation of the Machiavellian layout of the configuration process, while the accompanying documentation provides only the vaguest outline of how the controls actually work.

For those happy to go with the defaults, perhaps things are not so bad. The install process consists of a batch of RPM packages along with setup instructions buried in a PDF, and once these have been followed the product is quickly up and running. An interface is even provided, but here there is little more than a summary of the product's version information and running status, as well as a button marked 'update'. Manual updating is also possible, with the definitions provided in the form of a self-extracting install file. On the test platform, this required several extra packages to be installed to support its extraction processes, but with these tasks carried out it worked without a hitch. The product was rendered fully operational, including the on-access scanning provided by the company's own technology, fairly easily.

It is only when the default settings must be changed that things become difficult. The configuration is not stored, as is standard in Unix/*Linux*, in a nice, humanly readable and easily adjusted configuration file. Instead, a database in the style of the *Windows* registry is used, and any changes must be passed into this using a dedicated configuration tool. This tool responds equally blankly to both accurate and errant attempts to render the lengthy, syntactically complex commands required. Frequent rechecking of the full list is a must to ensure the proper changes have been made, while documentation of the numerical codes representing such options as on-detection actions seems non-existent.

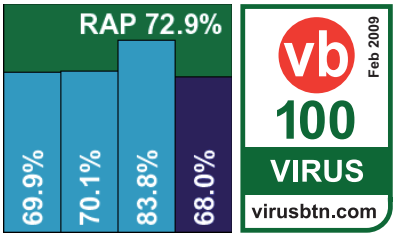
With the required tweaks assumed to have been made, the process of running command-line scans is a little less arduous, but by no means straightforward, and is similarly lacking in any form of feedback from the product. An option was found which would at least retain control of the command line, returning it when the scan completed, which enabled speed tests and monitoring of progress without recourse to checking the logging. This took the form of complex, barely readable output via the syslog facility, and in the main proved sufficiently usable to produce the required results.

In the RAP tests, eccentric and uneven figures hinted at a possible error in the multi-stage process of extracting information from the multi-record-per-line confusion of the logs, and a retry did produce different, but similarly erratic, results. In a slight bending of the VB100 'three attempts' rule, the scan was run multiple times and detections for each scan, varying by up to 10% each time, merged together to produce the final figures displayed here. It seems more than likely that this may not reflect the true detection capabilities of the product – for which I can only apologize to the vendor, but it was the best that could be achieved under the trying circumstances. If a more accurate way of procuring results can be found, we will strive to achieve it and update these figures – watch this space.

On a happier note, scanning speeds were pretty good and detection rates in the standard sets very good, with no difficulty handling the WildList and no false positives; a VB100 is duly awarded.

VirusBuster SambaShield 1.2.018-1.2.1.7

ItW	100.00%	**Polymorphic**	81.43%
ItW (o/a)	100.00%	**Trojans**	61.17%
Worms & bots	99.96%	**False positives**	0



The final product on the test bench brings what I had expected, based on past experience, to be the main rival to the *dazuko* setup in terms of on-access file-hooking: the Samba vfs object. *VirusBuster*'s product provides a series of .tgz files with an install script, making the installation reasonably straightforward despite a few rather vague passages of text. The setup of the *Samba* protection must be done manually, with instructions provided, but a slight inaccuracy in the guidelines led to the *Samba* share in question being rendered completely inaccessible – safe from malware perhaps, but hardly the ticket. A small tweak soon had things operational however, and testing continued.

Scanning speeds were fairly decent, and overheads pretty good too, but detection rates lagged a little behind this

month's very strong field. Logging proved a particular issue, with complex multi-line logs not the most comfortable to parse – for *Linux* administrators with large amounts of text-based data to handle, such inelegancies weigh heavily, just as they do for testers. However, after a period of hair-pulling and text-mangling, usable results were obtained, showing some fairly decent scores in the RAP sets, which improved considerably when the non-default 'grayware' option was enabled. Moving on to the VB100 certification requirements, the WildList was once again covered thoroughly and with no false alarms generated in the clean test set, *VirusBuster* is awarded a VB100.

CONCLUSIONS

The *Linux* test is always a bit of a roller-coaster of delight and despair, and here both highs and lows were very much in evidence. Some products were well designed, sensibly laid out and clearly documented, while others seemed to go out of their way to be obtuse, awkward and uncooperative. Nevertheless, most were somehow wrangled into line and useful results obtained, with on-access problems – once a major difficulty under *Linux* and the cause of many failures – put firmly in the past. All the products here managed to provide their on-access functionality smoothly and, for the most part, efficiently.

Performance is a significant issue, and in past *Linux* tests we have seen wide variations in scanning times and overheads, particularly between products using the same method to handle file access hooking. However, this again seems a thing of the past, with the gap between the faster and slower products narrowing.

Of course, the speed results depend a lot on the depth of scanning on offer and on the variety of file and archive types being analysed, and this is why we provide the additional archive table and scanning speeds for both default and full modes. With command-line products we often expect the default setting either to be everything off or everything on, but the trend was bucked this month with a wide variety of default settings, from thorough scanning with automatic disinfection or removal, through to fast and light scanning with reporting only. What guidance was available, in the form of usage notes, man pages and full manuals, generally required thorough reading before any assumptions could be made about the product's operation.

The limited number of updates to the WildList made for a fairly easy month for our small field of competitors, with none of them in any way troubled by the contents of the list. Hopes of a full set of VB100 awards were dashed, however, by a couple of unlucky false positives from otherwise high-performing products.

Of course, of interest to many this month will be the first set of results from our RAP testing. These figures conformed largely with our expectations. The extent of the decrease in detection seen in the 'week +1' results gives a reasonable indication of which products are using strong heuristic and generic detection, and which rely more heavily on fast response to new sightings.

The RAP results include some anomalous figures, not least in the earliest batch of samples, which many products fared less well against than those seen more recently. One explanation may be the coincidence of public holidays with that week of sample gathering, and the possibility that depleted labs may not have processed quite as much as usual. Other problems included a couple of products with logging and classification complications, which highlight the need to further refine the system and to define the rules of engagement more precisely. Further improvements are also planned to the back end of the set-up, including sample selection, automated validation procedures and so on, and we hope that the build-up of results over time will show some interesting trends and patterns.

Normally in this spot it would be my duty to point out that this type of static scanning does not fully reflect the overall capabilities of the product, as additional functionality may provide an extra layer of protection. On the desktop this is, of course, true, with a range of additional barriers being added to the latest generations of products. On file servers and at gateways however, the static scanning engine remains king, and detection rates, along with speed, usability and other factors looked at here, will continue to be the prime measure of product performance. We hope the latest addition to the information provided here helps give our readers some deeper insight into these factors.

Technical details

All products were tested on identical systems with *AMD Athlon64* X2 Dual Core 5200+ processors, 2 GB RAM, dual 80GB and 400GB hard drives, running *Red Hat Enterprise Linux 5.2*.

On-access tests were run from an AMD Sempron 3000+, 1.79GHz client with 512MB RAM, running *Microsoft Windows XP SP3*, connected via 100MB/s networking and *Samba* version 3.0.28-1.

Any developers interested in submitting products for VB's comparative reviews should contact john.hawes@virusbtn.com. The current schedule for the publication of VB comparative reviews can be found at http://www.virusbtn.com/vb100/about/schedule.xml.

END NOTES & NEWS

Black Hat DC 2009 takes place 16–19 February 2009 in Washington, DC, USA. Online registration is now open (onsite registration rates apply from 14 February). 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. Online registration is now open (onsite registration rates apply from 14 March). 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. 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 opens 1 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/.

Hacker Halted 2009 takes place in Miami, FL, USA, 23–24 September 2009. See http://www.hackerhalted.com/usa.

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

NEWS & EVENTS

FIRST OFFENDER PUNISHED UNDER NEW ZEALAND ANTI-SPAM LAW

A New Zealand man has become the first person to be prosecuted for spamming under the country's 2007 Unsolicited Electronic Messages Act.

Lance Atkinson confessed to having organized a group of people trading under the name Sancash which was responsible for sending more than two million messages over a three-month period at the end of 2007. The messages advertised the products of a company which paid Atkinson a commission of between 53% and 56% of the purchase price of each item sold. Atkinson now faces a fine of NZ$100,000 (approximately £35,000).

Meanwhile, the National Communications Commission of Taiwan has drafted a bill that will allow recipients of spam to claim damages from spammers of between NT$500 and NT$2,000 per email. Statistics from the Taiwan Internet Association indicate that the number of Internet users in Taiwan exceeded 10 million in December and that the country's users receive an average of 29 spam messages per day. The draft will be submitted for approval next month.

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. See http://www.maawg.org/.

The MIT Spam conference 2009 takes place 26–27 March 2009 in Boston, MA, USA. For details and a call for papers see http://projects.csail.mit.edu/spamconf/SC2009-cfp.html.

The Counter-eCrime Operations Summit will be held 12–14 May 2009 in Barcelona. See http://www.antiphishing.org/.

The sixth Conference on Email and Anti-Spam (CEAS) will be held 16–17 July 2009 in Mountain View, CA, USA. See http://www.ceas.cc/.

COMPARATIVE REVIEW – PROLOGUE

ANTI-SPAM TESTING: FREQUENTLY GIVEN ANSWERS

Martijn Grooten

Last month I outlined the proposed test set-up for *VB*'s comparative anti-spam tests (see *VB*, January 2009, p.S1). Following the publication of the article we received a lot of feedback from vendors, researchers and customers alike. It is great to see so much interest in our tests, and even better to receive constructive comments and suggestions.

Of course, several queries have been raised about our proposals – this article answers three of the most commonly asked questions.

FILTERING HERE OR FILTERING THERE?

For customers who want to buy an anti-spam solution for their incoming email – generally embedded into a larger email suite – the choice is not simply one of comparing different vendors. They could choose a product that can be embedded into an existing mail server, or one that is a mail server in itself – in which case there is a further choice between products that come with their own hardware and products that need to be installed on an existing operating system. But there are also products where both email filtering and mail hosting take place at the vendor's server; such products, labelled 'Software as a Service' (SaaS), are becoming increasingly popular.

Many vendors have asked whether our test will be able to accommodate SaaS products. The answer is yes – since the two major test criteria, the false positive rate and the false negative rate, can be measured for each of the product types mentioned above and can also be compared amongst them.

Of course, there are other metrics that describe a product's performance, not all of which apply to all types of product. For instance, the average and maximum CPU usage of a product are important measures for those that need to be installed on the user's machine, but are of little or no importance for products that provide their own hardware or are hosted externally. As a result, we aim to measure these aspects of performance in products for which they

are relevant, but the measurements will not be part of the certification procedure.

LEARN TO GET BETTER

One of the properties of spam is that it is indiscriminate; one of the properties of ham is that it is not. A classic example is that of pharmaceutical companies – whose staff might have legitimate reasons for sending and receiving email concerning body-part-enhancing products, but may find such email content blocked by spam filters. Many spam filters, however, are not indiscriminate and can learn from feedback provided by the end-user. Some filters even rely solely on user feedback: by default, all email messages have a spam probability of 0.5 and by combining user feedback with, among other things, Bayesian and Markovian methods, the product will 'learn' what kind of emails are unwanted and should be filtered as spam.

However, for a number of reasons, we have decided to test all products out-of-the-box using their default settings and not to provide filters with any user feedback.

Firstly, providing feedback would complicate our test set-up. In the real world, feedback is delivered to a learning filter whenever the user reads their email, which is generally multiple times during the day. In our set-up, the 'golden standard' will be decided upon by our end-users at their leisure (meaning they do not have to make classification decisions under pressure, thus minimizing mistakes), so our feedback would not be representative of a real-world situation.

Secondly, the performance of a learning filter as perceived by the user will not depend solely on its ability to learn from user feedback, but at least as much on the quality of the feedback given. If deleting a message is easier/less time-consuming than reporting it as spam, users might just delete unwanted email from their inbox; messages that are wanted but do not need to be saved might be read in, but not retrieved from the junk mail folder; the 'mark as spam' button might be used as a convenient way of unsubscribing to mailing lists. The quality of the feedback given thus depends on the end-user's understanding of how to provide feedback, as well as the ease with which they can provide it. We do not currently believe we can test this in a fair and comparable way. Of course, we will continue to look for possible ways to include learning filters in our tests.

PROACTIVE FILTERING METHODS

A wide range of anti-spam measures are based on the content of the email or the context in which it was sent, and most filters use a combination of such measures. However, many filters also take a more proactive approach, where they try to frustrate the spammers, for instance by delaying their response to SMTP commands ('tarpitting') or by temporarily refusing email from unknown or unverifiable sources ('greylisting').

Such methods assume that legitimate senders will keep trying to get the message delivered, while many spammers will give up: apart from the fact that mail agents used by spammers are often badly configured, the spammers' economic model is based on being able to deliver a large volume of messages in a short period of time and it will generally not be viable for them to keep trying.

From the receivers' point of view, these methods are as good as any other to stop spam, but with two major drawbacks. Firstly, greylisting could cause significant delays to the delivery of some legitimate email, which could be disadvantageous in a business environment. Secondly, any such proactive anti-spam method could result in false positives that are impossible to trace – which, again, is undesirable for a business that wants to be able to view all incoming emails, even those classified initially as spam.

Such methods also cause a problem for the tester: the efficiency of an anti-spam method can only be tested if both the spam catch rate and the false positive rate can be measured. This is impossible with proactive methods, since these 'block' email before it is sent. This is one of the reasons why we will not be able to test against such methods with the set-up that uses our own email stream.

We realize that this will be a problem for products that make extensive use of these methods, and as a compromise we are looking for ways to expose all products to the email stream sent to a spam trap, which is (almost) guaranteed to be spam only. Of course, this will not solve the problem of testing for false positives.

WHAT HAPPENS NEXT?

We will be running a trial test this month. During the trial it is possible (indeed probable) that the test configuration will be changed. The results, therefore, may not be representative of those that would have been derived from a real test. For this reason, we intend to publish the results of the trial without specifying which products achieved them.

The first real test will start towards the end of March; vendors and developers will be notified in due course of the deadline and conditions for submitting a product.

As always, we welcome comments, criticism and suggestions – and will continue to do so once the tests are up and running. Our goal is to run tests in which products are compared in a fair way, and which will produce results that are useful to end-users. Any suggestions for better ways in which our tests could achieve these goals will be given serious consideration (please email martijn.grooten@virusbtn.com).