MARCH 2010

virus BULLETIN

Fighting malware and spam

CONTENTS

IN THIS ISSUE

EAGLE EYES

If a file contains no code, can it be executed? Can arithmetic operations be malicious? In W32/Lerock we have a file that contains no code, and no data in any meaningful sense. All it contains is a block of relocation items. Peter Ferrie untangles the mystery.
page 4

BACK DOOR SURPRISE

New BackDoor.Tdss rootkits are sophisticated pieces of malware. Alexey Tkachenko and Artem Baranov detail the BackDoor.Tdss.565 rootkit – which presented surprises within minutes of the start of its analysis.
page 6

VBSPAM CERTIFICATION

VB is delighted to report that, for the first time, all 16 of the products in this month's test achieved a VBSpam award. Martijn Grooten has the details.
page 24

ISSN 1749-7027

'We're wasting more time than ever dealing with malware that is more hostile than ever.'

John Levine, Taughannock Networks

WHY FLASH WEB PAGES ARE LIKE COLLATERALIZED DEBT OBLIGATIONS

For the past few weeks I've been trying to track down a bot that has been sending spam from my home DSL line. I know it's there, because it has got me onto several blacklists such as the CBL (http://cbl.abuseat.org/). Like most home users, I send and receive my mail via a server somewhere else, so the blacklist entry doesn't affect me directly, but I want to be a good citizen, and besides, it's embarrassing for 'Mr. Spam Expert' to be on a blacklist.

On the LAN behind the DSL router are VoIP phones, a printer, a Mac, and a laptop running *FreeBSD*, but the prime suspects are two *Windows 7* laptops: mine and my daughter's. Multiple anti-malware programs on both *Windows* boxes all swear that both machines are clean.

The phone company gave me a combination DSL modem-router-access point, managed through web pages, telnet, and even FTP. I've turned off one computer or another to see if the spam would stop, and although I can't sniff the switched wired LAN, I sniffed the wi-fi where both *Windows* boxes are, and saw no port 25 mail traffic, even when people were getting spam. Poking around in the router, I found its internal logs, with mysterious UPNP port forwarding entries from my daughter's machine. Aha! So I dug out the *Windows 7* install disk, wiped the laptop clean, reinstalled from scratch, and the spam still didn't stop. Now I'm trying netstat, and it looks like the other one's infected, too.

My main thought during this process has been: 'what a *phenomenal* waste of time'.

Since the first computer virus hopped onto a floppy disk about two decades ago, how much progress have we made against malware on our computers? To put it baldly, less than none. We're wasting more time than ever dealing with malware that is more hostile than ever. In the good old days, a virus might have drawn odd squiggles in the corner of your screen. Now it sends floods of porn spam while siphoning money from your bank account. What are we doing wrong? Our fundamental attitude toward software is screwed up.

In the past decade, the world has learned the hard way about the perils of financial innovation. Banks broke free from traditional regulation and innovated like crazy, with consequences that we now all know. It might have seemed like a good idea at the time to invent multiple tranches of derivative securities based on no-doc mortgages on shoddily built houses hours away from any jobs, but now we know that 'innovation' mostly meant very large levels of unknown risk, with the consequences falling on someone other than the innovator when they screw up. Does this remind you of anything?

Contrary to popular belief, there's no secret to writing very reliable software. Computer-controlled space probes operate reliably for years, billions of miles from the nearest repair depot. Large airline and bank systems are equally reliable; one system running *IBM*'s TPF has run continuously for ten years, through multiple hardware and software upgrades. Reliability like that comes from having a very different attitude toward software: nothing changes unless there's a very good reason to change it, nothing goes into the system without being thoroughly reviewed, and nothing goes in just because it's cute and blinky.

The time we spend dealing with malware and its consequences is a dead weight on computer users, which is notably not charged back to the people who made the vulnerable software. When I look at my word processor or my web browser, I see about 100,000 bells and whistles, 99,900 of which I have never used and never will. If you ask a user 'would you like feature X?', the answer is always 'yes'. But ask the question: 'do you want feature X if it's likely to mean that you waste days deworming your computer, or arguing with your bank to get stolen money back, or desperately hoping that you backed up the data you lost in crashes it caused?', the answer is of course 'no'.

Banks generally work just fine doing what they've done all along, and for most computer users, their computers work just fine doing what they've done all along, too. If we pushed back and said 'no' to glitz, and 'yes' to conservative design, imagine how much better off we'd all be.

NEWS

HMRC ADDRESSES SECURITY

After catastrophic data losses in November 2007 (including the personal details of all UK families with a child under the age of 16 – affecting some 25 million people), Her Majesty's Revenue and Customs (HMRC) is now putting IT security firmly at the heart of its business strategy.

HMRC's 85,000 staff will undergo re-training in a bid to prevent future data losses, and secure computing will be included in the performance objectives of every employee.

The organization aims to improve its staff's security consciousness and data-handling behaviour, providing a data security rule book, data security workshops and a dedicated security zone on its intranet. Senior management will be encouraged to champion security, and 'data guardians' are to be appointed in each of HMRC's business units. *VB* applauds the initiative and only wonders why it has taken so long to be put into effect.

MICROSOFT TACKLES WALEDAC

A court order was obtained by *Microsoft* last month to force the takedown of close to 300 Internet domains associated with the Waledac botnet. The court order, obtained by *Microsoft* as part of its 'Operation b49', forces *VeriSign* to cut off 277 domains involved in the command and control of Waledac's network of compromised machines.

Waledac is believed to have infected hundreds of thousands of machines around the world and has been a major source of spam – *Microsoft* found that, in an 18-day period in December 2009, approximately 651 million spam emails attributable to Waledac were directed to *Hotmail* accounts alone.

Further countermeasures have been taken by *Microsoft* to downgrade the remaining peer-to-peer command and control communication within the botnet, and the company reports that it has effectively shut down connections to the vast majority of Waledac-infected computers.

Microsoft hints that more such legal and industry operations are in the pipeline.

MCAFEE GROWTH PLAN UNVEILED

McAfee plans to acquire three to four companies every year to help drive its growth, according to chief executive David DeWalt, who says that he sees small- and medium-sized acquisitions as the way forward for the company. He also quashed suggestions by industry analysts that *McAfee* itself would make an ideal takeover target for giants such as *Hewlett-Packard* and *IBM*, saying simply: 'We are not for sale.'

Prevalence Table – January 2010[1]

Malware	Type	%
Adware-misc	Adware	11.24%
FakeAlert/Renos	Rogue AV	11.01%
Autorun	Worm	10.46%
Conficker/Downadup	Worm	6.87%
VB	Worm	5.35%
OnlineGames	Trojan	3.88%
WinWebSec	Rogue AV	3.03%
HackTool	PU	2.99%
Agent	Trojan	2.95%
Heuristic/generic	Virus/worm	2.82%
Virut	Virus	2.70%
Downloader-misc	Trojan	2.18%
Delf	Trojan	2.17%
Istbar/Swizzor	Trojan	2.14%
Hupigon	Trojan	1.78%
PDF	Exploit	1.73%
Zbot	Trojan	1.69%
Inject	Trojan	1.65%
Alureon	Trojan	1.52%
Navipromo/Skintrim	Trojan	1.49%
Wintrim	Trojan	1.43%
Small	Trojan	1.43%
Exploit-misc	Exploit	1.22%
Crack	PU	1.12%
Bifrose/Pakes	Trojan	0.98%
Sality	Virus	0.96%
Crypt	Trojan	0.91%
Heuristic/generic	Trojan	0.91%
Ircbot	Worm	0.77%
Backdoor-misc	Trojan	0.73%
KillAV	Trojan	0.72%
Allaple	Worm	0.71%
Others[2]		8.69%
Total		100.00%

[1] This month's prevalence figures are compiled from desktop-level detections.

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

MALWARE ANALYSIS 1

DOIN' THE EAGLE ROCK

Peter Ferrie
Microsoft, USA

If a file contains no code, can it be executed? Can arithmetic operations be malicious? Here we have a file that contains no code, and no data in any meaningful sense. All it contains is a block of relocation items, and all relocation items do is cause a value to be added to locations in the image. So, nothing but relocation items – and yet it also contains W32/Lerock.

Lerock is written by the same virus author as W32/Fooper (see *VB*, January 2010, p.4), and behaves in the same way at a high level, but at a lower level it differs in an interesting way.

EXCEPTIONAL BEHAVIOUR

Like Fooper, the virus begins by walking the Structured Exception Handler chain to find the topmost handler, and at the same time registers a new exception handler which points to the host entrypoint. Once it has found the topmost handler, the virus uses the resulting pointer as the starting location in memory for a search for the MZ and PE headers of kernel32.dll. Once it has found the headers, the virus parses the export table to find the APIs that it needs for infection.

The first problem in Lerock's code is identical to the first bug in Fooper's code: in *Windows Vista* and later, the topmost handler points into ntdll.dll rather than kernel32.dll. As a result, the virus crashes on these platforms, because it assumes that the APIs it needs for infection will be found, and falls off the end of a buffer because they do not exist.

HAPI HAPI, JOY JOY

If the virus finds the PE header for kernel32.dll, then it resolves the required APIs. It uses hashes instead of names, but the hashes are sorted alphabetically according to the strings they represent. This means that the export table only needs to be parsed once for all of the APIs, rather than parsing once for each API (as is common in some other viruses). Each API address is placed on the stack for easy access, but because stacks move downwards in memory, the API addresses end up in reverse order in memory.

LET'S DO THE TWIST

After retrieving the API addresses from kernel32.dll, the virus initializes its Random Number Generator (RNG). Like Fooper, Lerock uses a complex RNG known as the 'Mersenne Twister'. In fact, the virus author has used this RNG in every virus for which he requires a source of random numbers.

The virus then allocates two blocks of memory: one to hold the intermediate encoding of the virus body, and the other to hold the fully encoded virus body. The virus decompresses a file header into the second block. The file header is compressed using a simple Run-Length Encoder algorithm. The header is for a *Windows* Portable Executable file, and it seems as though the intention was to produce the smallest possible header that can still be executed on *Windows*. There are overlapping sections, and 'unnecessary' fields have been removed. The virus then allocates a third block of memory, which will hold a copy of the unencoded virus body.

The virus searches for zeroes within the unencoded memory block and keeps a count of them. The zeroes will be skipped during the encoding process, which is the next step.

RELOCATION ALLOWANCE

The virus chooses randomly among the bytes in its body until it finds one whose value is not zero. For each such byte that is found, the virus stores the RVA of the byte within the encoding memory block, along with a relocation item whose type specifies that the top 16 bits of the delta should be applied to the value. The result of this is to add one to the value. The reason why this occurs is as follows:

The virus uses a file whose ImageBase field is zero in the PE header. This is not a valid loading address in *Windows*, so when *Windows* encounters such a file, it will relocate the image (with the exception of *Windows NT*, which does not support the relocation of .exe files at all). However, the location to which the relocation occurs is different for the two major *Windows* code-bases. *Windows NT*-based versions of *Windows* (specifically, *Windows 2000* and later) relocate images to 0x10000. *Windows 95*-based versions (*Windows 9x/Me*) relocate images to 0x400000. It is the *Windows NT*-based style of behaviour that the virus requires.

When relocation occurs, *Windows* calculates the delta value to apply. This value is calculated by subtracting the old loading from the new loading address (this can be a negative value if the image loads to a lower address than it requested). In this case, the new loading address is 0x10000, and the old loading address is 0, so the delta is also 0x10000, or to be more explicit, 0x00010000. Thus, the top 16 bits of the delta are 0x0001. It is this trick that allows the virus to adjust the value by one.

The virus decreases the value of the byte within the unencoded memory block. If the value reaches zero, then the virus decreases the number of bytes left to process. The virus also increases the corresponding value in the intermediate encoding memory block.

At this point, the virus decides randomly if it should apply special relocation items to the surrounding values, and, if so, what type of items to apply. The virus can produce a relocation item that adds 0x40 to any byte that is in the location one byte after the current position, but it has a side effect (not all of the bits are maintained) on three of the four bytes beginning at the current position, so the virus selects this type only if the next three bytes are still zero. The virus subtracts 0x40 from the value of the byte within the unencoded memory block. If the value reaches zero, then the virus decreases the number of bytes left to process.

The virus can also produce a relocation item that adds 0x20 to any byte that is in the location 13 bytes after the current position, but it has the same side effect as above, on a much larger scale (10 out of 16 bytes are affected), so the virus selects this type only if those 10 bytes are still zero. The virus subtracts 0x20 from the value of the byte within the unencoded memory block. If the value reaches zero, then it decreases the number of bytes left to process.

This is where the intermediate encoding memory block comes into play. It is a representation of the relocation items that have been applied at the current moment in time. The buffer begins by containing all zeroes, and the values are increased as the relocation items are applied. The ultimate aim is to reduce all of the original non-zero bytes to zero, thus avoiding the need to have any code in the file. All that is left is an empty section.

The encoding process repeats until all of the non-zero bytes have been encoded. The random ordering and type selection of the relocation items produces an essentially polymorphic representation of the virus body. Once the encoding process has completed, the virus creates a file called 'rel.exe', places the size information into the section header, writes the encoded body, then runs the resulting file. Finally, it transfers control to the host.

DROPPING YOUR BUNDLE

The dropped file begins by walking the Structured Exception Handler chain to find the topmost handler, and at the same time registers a new exception handler, which points to the host entrypoint. As above, the code locates kernel32.dll in order to resolve the APIs that it needs for replication. Unlike the W32/Fooper, this virus uses only Unicode-based APIs, since the *Windows* code base that it requires is also Unicode-based.

After retrieving the API addresses from kernel32.dll, the virus attempts to load 'sfc_os.dll'. If that attempt fails, then it attempts to load 'sfc.dll'. If either of these attempts succeed, then the virus resolves the SfcIsFileProtected() API. The reason the virus attempts to load both DLLs is that the API resolver in the virus code does not support import forwarding. The problem with import forwarding is that, while the API name exists in the DLL, the corresponding API address does not. If a resolver is not aware of import forwarding, then it will retrieve the address of a string instead of the address of the code. In the case of the SfcIsFileProtected() API, the API is forwarded in *Windows XP* and later, from sfc.dll to sfc_os.dll. Interestingly, the virus supports the case where neither DLL is present on the system, even though that can occur only on older platforms – which it does not support.

The virus then searches for files in the current directory and all subdirectories, using a linked list instead of a recursive function. This is simply because the code is based on existing viruses by the same author – this virus does not infect DLLs, so the stack size is not an issue. The virus avoids any directory that begins with a '.'. This is intended to skip the '.' and '..' directories, but in *Windows NT* and later, directories can legitimately begin with this character if other characters follow. As a result, such directories will also be skipped.

FILTRATION SYSTEM

Files are examined for their potential to be infected, regardless of their suffix, and will be infected if they pass a strict set of filters. The first of these is the support for the System File Checker that exists in *Windows 2000* and later.

The remaining filters include the condition that the file being examined must be a *Windows* Portable Executable file, a character mode or GUI application for the *Intel* 386+ CPU, not a DLL, that the file must have no digital certificates, and that it must not have any bytes outside of the image.

TOUCH AND GO

When a file is found that meets the infection criteria, it will be infected. The virus resizes the file by a random amount in the range of 4–6KB in addition to the size of the virus. This data will exist outside of the image, and serves as the infection marker.

If relocation data is present at the end of the file, the virus will move the data to a larger offset in the file, and place its code in the gap that has been created. If no relocation data is present at the end of the file, the virus code will be placed here. The virus checks for the presence of relocation data by checking a flag in the PE header. However, this method is unreliable because *Windows* ignores this flag, and relies instead on the base relocation table data directory entry.

The virus increases the physical size of the last section by the size of the virus code, then aligns the result. If the virtual size of the last section is less than its new physical size, then the virus sets the virtual size to be equal to the physical size,

and increases and aligns the size of the image to compensate for the change. It also changes the attributes of the last section to include the executable and writable bits. The executable bit is set in order to allow the program to run if DEP is enabled, and the writable bit is set because the RNG writes some data into variables within the virus body.

The virus alters the host entrypoint to point to the last section, and changes the original entrypoint to a virtual address prior to storing the value within the virus body. This will prevent the host from executing later, if it is built to take advantage of Address Space Layout Randomization (ASLR). However, it does not prevent the virus from infecting files first. The lack of ASLR support might be considered a bug but for the fact that ASLR was only introduced in *Windows Vista*, which the virus does not support. What is strange, though, is that changing the entrypoint affects DLLs in the same way. Thus, if an infected DLL is relocated because of an address conflict, then it, too, will fail to run.

APPENDICITIS

After setting the entrypoint, the virus appends the dropper code. Once the infection is complete, the virus will calculate a new file checksum, if one existed previously, before continuing to search for more files.

Once the file searching has finished, the virus will allow the host code to execute by forcing an exception to occur. This technique appears a number of times in the virus code, and is an elegant way to reduce the code size, as well as functioning as an effective anti-debugging method.

Since the virus has protected itself against errors by installing a Structured Exception Handler, the simulation of an error condition results in the execution of a common block of code to exit a routine. This avoids the need for separate handlers for successful and unsuccessful code completion.

CONCLUSION

The virus author called this technique 'virtual code', which is quite an accurate description. However, the technique lends itself to simple detection by anti-virus software, given the randomly ordered relocation items that are applied multiple times to bytes in an empty section. Of course, future virus writers might try to bypass detection by ordering the relocation items sequentially (which would make it less suspicious, but reduce the polymorphism at the same time). Alternatively, they might fill the section with legitimate-looking code and transform that instead (which would make it less suspicious, but potentially require even more relocation items). However, what remains is still a set of relocation items that are applied multiple times to bytes in a section, and there's no getting around that one.

MALWARE ANALYSIS 2

BACKDOOR.TDSS.565 AND ITS MODIFICATIONS (AKA TDL3)

Alexey Tkachenko, Artem Baranov
Doctor Web, Russia

The Backdoor.Tdss.565 rootkit presented us with surprises within minutes of embarking on its analysis. For instance, its non-typical method of injection into a system process during installation was completely unexpected. Though documented, the method has never before been implemented in any known virus, and therefore it allows the rootkit to bypass most behaviour blockers, install its driver and remain undetected.

The installation process continues in kernel mode. The rootkit searches through the stack of devices responsible for interaction with the system disk to determine which driver it will infect. The choice depends on the hardware configuration. If the system disk uses the IDE interface, it will pick out atapi.sys; in other cases it may be iastor.sys. There are other rootkits that infect file system and network drivers or even the system kernel to ensure their automatic launch (e.g. BackDoor.Bulknet.415, Win32.Ntldrbot, Trojan.Spambot.2436 and others), so this case is not an exception. Note that the file size remains unchanged because the malicious code is written over a part of the file's resources section. In fact, the piece of code only occupies 896 bytes (in later versions this is reduced to 481 bytes) and it loads the main body of the rootkit. At the same time it changes the entry point, sets the driver signature link to null, and recalculates the file's hash sum. Addresses of the API functions used by the loader for infection are located in its body as RVAs. This both reduces the size of the loader and complicates analysis of the infected driver in the system that uses a different version of the kernel.

Next, the malware assesses the available disk space and utilizes a small part (24,064 bytes) from the end of the disk for storage of the rootkit's main body – or, more precisely, for storage of the part of the driver that performs the installation saved as binary data instead of an executable image. The block starts with the 'TDL3' marker, followed by 896 bytes of the genuine resource code of the infected driver. The malware also creates a separate virtual drive where its user-mode components and configuration file are located. It seems likely that this trick was inspired by BackDoor.Maxplus, which also created a virtual disk to deploy its components in the system. The process will be described in more detail later in this article.

One of the rootkit's later versions, BackDoor.Tdss.1030, stores original resources data and its body on the hidden

```
.rsrc:00026780                  push    ebp
.rsrc:00026781                  mov     ebp, esp
.rsrc:00026783                  sub     esp, 160h
.rsrc:00026789                  call    $+5
.rsrc:00026789
.rsrc:0002678E                  pop     eax
.rsrc:0002678F                  sub     eax, 0F9A82A0Bh
.rsrc:00026794                  mov     [ebp+var_C], eax
.rsrc:00026797                  mov     eax, [ebp+var_C]
.rsrc:0002679A                  add     eax, 350h
.rsrc:0002679F                  add     eax, 0F9A829FDh
.rsrc:000267A4                  mov     [ebp+pParamBlock], eax
.rsrc:000267A7                  cmp     [ebp+RegistryPath], 1
.rsrc:000267AB                  jbe     jIsFsChangeOccur ; atapi init, not jmp
.rsrc:000267AB
.rsrc:000267B1                  mov     eax, [ebp+DriverObject]
.rsrc:000267B4                  mov     eax, [eax+DRIVER_OBJECT.DriverSection]
.rsrc:000267B7                  mov     eax, [eax]
.rsrc:000267B9                  mov     [ebp+PsLoadedModuleList], eax
.rsrc:000267B9
.rsrc:000267BC
.rsrc:000267BC  jNextDriverLdrEntry:                    ; CODE XREF: DriverEntry+4F↓j
.rsrc:000267BC                  mov     eax, [ebp+PsLoadedModuleList]
.rsrc:000267BF                  movzx   eax, [eax+KLDR_DATA_TABLE_ENTRY.LoadCount]
.rsrc:000267C3                  test    eax, eax
.rsrc:000267C5                  jz      short jPsLoadedModuleListWasFound
```

Figure 1: The entry point of atapi.sys compromised by BackDoor.Tdss.565.

encrypted drive in rsrc.dat and tdl files respectively, which significantly simplifies its updating.

Upon completion of the installation the driver returns a STATUS_SECRET_TOO_LONG (0xC0000154) error which informs user-mode components (http://vms.drweb.com/search/?q=BackDoor.Tdss.565) that installation has completed successfully and causes the system to unload the driver that is no longer used by the rootkit.

THE LOADER

The viral loader starts working along with the infected driver. As mentioned above, its main task is to load the rootkit's body stored at the 'end' of the hard drive. Since the loader starts working when the hard drive port driver is loaded by the kernel, it still can't work with the disk or the file system. This is why it first registers a notification routine for the creation of FS (FileSystem) control device objects, and only then does it load the rootkit's body. Early versions of the malware used the IoRegisterFsRegistrationChange function for this purpose, while the later ones resort to the temporary interception of the victim's IRP_MJ_DEVICE_CONTROL in DRIVER_OBJECT where the dispatcher waits for a certain request from the file system. Remarkably, in both cases the entry point of the infected driver is used both to start the original DriverEntry as well as for the FS standby (Figure 1).

Let's assume that atapi.sys is the compromised driver.

Now let's take a closer look at how the BackDoor.Tdss.565 loader works. Once it has gained control, it will go over the sections table of its media and modify it to make detection of the initialization section more complicated: it nulls the IMAGE_SCN_MEM_DISCARDABLE bit of each section, and replaces the first byte of a name with zero if it is INIT. It also reserves an auxiliary data structure to save the pointer to the atapi driver object. After that it uses the CDO (Control Device Object) to register the FS creation notification sent to the kernel.

As the file system request is received, the second part of the loader is started. It checks all object-devices of the port driver (e.g. '\Device\IdePort0', '\Device\IdeDevicePOT0L0-3') and uses the disk offset placed in its body during installation to read the rootkit's body. Although using the ordinary ZwOpenFile and ZwReadFile functions for this purpose seems rather unsophisticated (as the malware has to check devices one by one), it allows the loader to remain compact and serves its purpose quite well. The TDL3 signature placed at the beginning of the data segment is used to verify that the reading has been successful (Figure 2). After that, the notification is deleted (IoUnregisterFsRegistrationChange) and control is transferred to the body of the rootkit.

```
x00FFFFD0  x000  54 44 4C 33 00 00 00 00 00 00 00 00 00 00 00 00  TDL3............
16 777 168 x010  00 00 01 00 10 00 00 00 18 00 00 80 00 00 00 00  ..........Ъ.....
           x020  00 00 00 00 00 00 00 00 00 00 01 00 01 00 00 00  ................
           x030  30 00 00 80 00 00 00 00 00 00 00 00 00 00 00 00  0..Ъ............
           x040  00 00 01 00 09 04 00 00 48 00 00 00 E0 67 01 00  ........H...ag..
           x050  7C 03 00 00 00 00 00 00 00 00 00 00 00 00 00 00  |...............
           x060  00 00 00 00 7C 03 34 00 00 00 56 00 53 00 5F 00  ....|.4...V.S._.
           x070  56 00 45 00 52 00 53 00 49 00 4F 00 4E 00 5F 00  V.E.R.S.I.O.N._.
           x080  49 00 4E 00 46 00 4F 00 00 00 00 00 BD 04 EF FE  I.N.F.O.....S.ю
           x090  00 00 01 00 01 00 05 00 88 15 28 0A 01 00 05 00  ........€.(.....
           x0A0  88 15 28 0A 3F 00 00 00 00 00 00 00 04 00 04 00  €.(.?...........
           x0B0  03 00 00 00 07 00 00 00 00 00 00 00 00 00 00 00  ................
           x0C0  DC 02 00 00 01 00 53 00 74 00 72 00 69 00 6E 00  b.....S.t.r.i.n.
           x0D0  67 00 46 00 69 00 6C 00 65 00 49 00 6E 00 66 00  g.F.i.l.e.I.n.f.
           x0E0  6F 00 00 00 B8 02 00 00 01 00 30 00 34 00 30 00  o...ё.....0.4.0.
           x0F0  39 00 30 00 34 00 42 00 30 00 00 00 4C 00 16 00  9.0.4.B.0...L...
           x100  01 00 43 00 6F 00 6D 00 70 00 61 00 6E 00 79 00  ..C.o.m.p.a.n.y.
           x110  4E 00 61 00 6D 00 65 00 00 00 00 00 4D 00 69 00  N.a.m.e.....M.i.
           x120  63 00 72 00 6F 00 73 00 6F 00 66 00 74 00 20 00  c.r.o.s.o.f.t...
           x130  43 00 6F 00 72 00 70 00 6F 00 72 00 61 00 74 00  C.o.r.p.o.r.a.t.
           x140  69 00 6F 00 6E 00 00 00 54 00 16 00 01 00 46 00  i.o.n...T.....F.
           x150  69 00 6C 00 65 00 44 00 65 00 73 00 63 00 72 00  i.l.e.D.e.s.c.r.
           x160  69 00 70 00 74 00 69 00 6F 00 6E 00 00 00 00 00  i.p.t.i.o.n.....
           x170  49 00 44 00 45 00 2F 00 41 00 54 00 41 00 50 00  I.D.E./.A.T.A.P.
           x180  49 00 20 00 50 00 6F 00 72 00 74 00 20 00 44 00  I...P.o.r.t...D.
           x190  72 00 69 00 76 00 65 00 72 00 00 00 62 00 21 00  r.i.v.e.r...b.!.
           x1A0  01 00 46 00 69 00 6C 00 65 00 56 00 65 00 72 00  ..F.i.l.e.V.e.r.
           x1B0  73 00 69 00 6F 00 6E 00 00 00 00 00 35 00 2E 00  s.i.o.n.....5...
           x1C0  31 00 2E 00 32 00 36 00 30 00 30 00 2E 00 35 00  1...2.6.0.0...5.
           x1D0  35 00 31 00 32 00 20 00 28 00 78 00 70 00 73 00  5.1.2...(.x.p.s.
           x1E0  70 00 2E 00 30 00 38 00 30 00 34 00 31 00 33 00  p...0.8.0.4.1.3.
           x1F0  2D 00 32 00 31 00 30 00 38 00 29 00 00 00 00 00  -.2.1.0.8.).....
```

Figure 2: The first sector of the rootkit's body located in end sectors of the hard drive.

THE ROOTKIT

An encrypted drive with its own file system is certainly among the most notable technical features of TDL3, but the mechanism used to hide an entire file or the part of an arbitrary disk sector on the port driver level is equally remarkable. No other known rootkit has implemented these concepts in full.

It is well known that the main feature of the *NT* virtual file system is the availability of all input-output devices on the descriptor layer where the key element is the file object created by the kernel and objects that represent the device. An application opens the descriptor for one channel, hard drive, volume or file, and different layers of the input-output devices stack participate in the interaction. The kernel only needs information about a request to start a corresponding dispatcher function.

The authors of the rootkit used a similar approach and implemented their file system to work on the level of the device object's port driver so that the virus mounts its FS to the device object.

The atapi driver creates several types of device object (Figure 3). The upper two are devices representing hard and CD drives, while the other two are controllers interacting with the mini-port driver implemented in *Windows XP* as a hybrid mix of a port and mini-port. To mount its hidden drive the rootkit chooses a device object with the FILE_DEVICE_CONTROLLER type.

An ordinary ('healthy') atapi driver uses only one IRP dispatch function to serve read/write requests – IRP_MJ_SCSI (IRP_MJ_INTERNAL_DEVICE_CONTROL). The client uses Srb and sends it to the disk device object. SUCCESS is always returned for Create/Close atapi requests, since the atapi doesn't use them. However, the Create operation is very important for the FSD (File System Driver) because it initializes FILE_OBJECT which is used for file operations.

The path to rootkit files located in the protected (hidden) area is as follows: \Device\Ide\IdePort1\mjqxtpex\, where

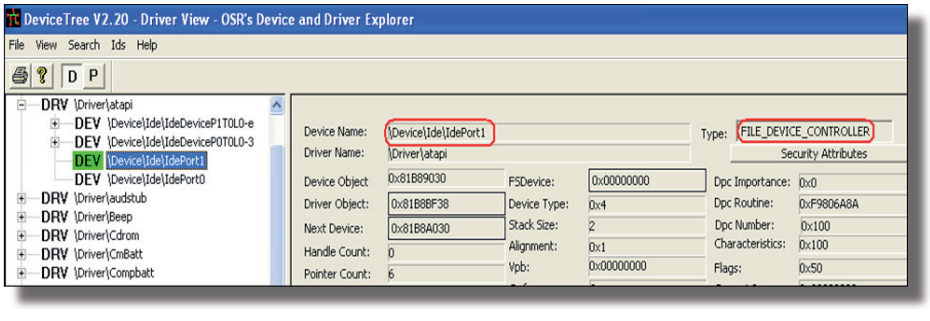


Figure 3: Devices created by atapi.sys.

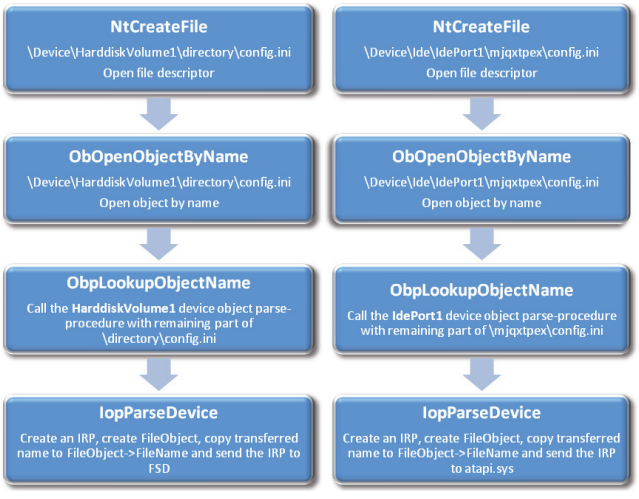


Figure 4: Opening a file on an ordinary disk drive (left) and on the hidden drive (right).

mjqxtpex is an eight-byte signature generated randomly at system start-up. The hidden drive is used by user-mode components of the rootkit to store files received from the Internet or to read their configuration.

The following are some full path examples:

\\?\globalroot\Device\Ide\IdePort1\mjqxtpex\tdlcmd.dll

\\?\globalroot\Device\Ide\IdePort1\mjqxtpex\tdlwsp.dll

\\?\globalroot\Device\Ide\IdePort1\mjqxtpex\config.ini

In order to understand how the rootkit works with its file system, let's take a look at a flow chart showing how a create request is normally processed (ntfs or fastfat), how \Device\HarddiskVolume1\directory\config.ini is opened on an ordinary drive, and how \Device\Ide\IdePort1\mjqxtpex\config.ini is accessed on the hidden drive (see Figure 4).

The rootkit has one shared dispatch function for all requests from atapi clients and user-mode components. Therefore it performs two important tasks:

- It hides data located in the protected area from atapi clients and provides clients with an original file as they try to read data from the disk.
- As with FSD, it handles create/close/query information requests for files from the protected area, as well as requests from the rootkit itself, such as to read a section of config.ini.

The rootkit replaces parameters in the dispatch functions pointer table

```
Dispatch routines:
[00] IRP_MJ_CREATE                      f9756b3a    atapi!PortPassThroughZeroUnusedBuffers+0x34
[01] IRP_MJ_CREATE_NAMED_PIPE           f9756b3a    atapi!PortPassThroughZeroUnusedBuffers+0x34
[02] IRP_MJ_CLOSE                       f9756b3a    atapi!PortPassThroughZeroUnusedBuffers+0x34
[03] IRP_MJ_READ                        f9756b3a    atapi!PortPassThroughZeroUnusedBuffers+0x34
[04] IRP_MJ_WRITE                       f9756b3a    atapi!PortPassThroughZeroUnusedBuffers+0x34
[05] IRP_MJ_QUERY_INFORMATION           f9756b3a    atapi!PortPassThroughZeroUnusedBuffers+0x34
[06] IRP_MJ_SET_INFORMATION             f9756b3a    atapi!PortPassThroughZeroUnusedBuffers+0x34
[07] IRP_MJ_QUERY_EA                    f9756b3a    atapi!PortPassThroughZeroUnusedBuffers+0x34
[08] IRP_MJ_SET_EA                      f9756b3a    atapi!PortPassThroughZeroUnusedBuffers+0x34
[09] IRP_MJ_FLUSH_BUFFERS               f9756b3a    atapi!PortPassThroughZeroUnusedBuffers+0x34
[0a] IRP_MJ_QUERY_VOLUME_INFORMATION    f9756b3a    atapi!PortPassThroughZeroUnusedBuffers+0x34
[0b] IRP_MJ_SET_VOLUME_INFORMATION      f9756b3a    atapi!PortPassThroughZeroUnusedBuffers+0x34
[0c] IRP_MJ_DIRECTORY_CONTROL           f9756b3a    atapi!PortPassThroughZeroUnusedBuffers+0x34
```

Figure 5: Windows XP SP3 atapi.sys interceptions.

as follows: it finds the end of the first section of the atapi.sys file in memory and writes the following template into the cave (the remaining free space in the section):

```
mov eax, ds:0FFDF0308h
jmp dword ptr [eax+0FCh]
```

In some cases the instructions can overwrite data in the adjacent section since there is no verification procedure. Therefore, interceptions are still directed to atapi.sys (Figure 5). This fools many anti-rootkits, so the malware remains undetected.

The rootkit utilizes a large structure for storage of all configuration information that may be required to perform its routines. The structure pointer is placed at 0xFFDF0308, i.e. a part of KUSER_SHARED_DATA is used. The request dispatcher is found at the +00FCh offset (invoked in the example above – jmp dword ptr [eax+0FCh]). Structures describing which sectors must be hidden and what should replace them are also stored there.

If an atapi client requests data from the protected drive, it will simply zero-fill it or replace it with original data. Let's take a look at the pseudo code showing how it works:

```
if( DeviceObject == ROOTKIT_PARAM_BLOCK.
AtapiBootRootkitDevObj &&
   IoStack->MajorFunction == IRP_MJ_SCSI &&
   IoStack->Parameters.Scsi.Srb->Function == SRB_
FUNCTION_EXECUTE_SCSI
)
{
if( RequestedStartSector + cSectors > ROOTKIT_PARAM_
BLOCK.HideAreaStartSector)
{
        if( IsRead )
        {
                Replace the completion function of
the current stack location with its own function
        }
        else if( IsWrite )
        {
                End operation and return an error
```

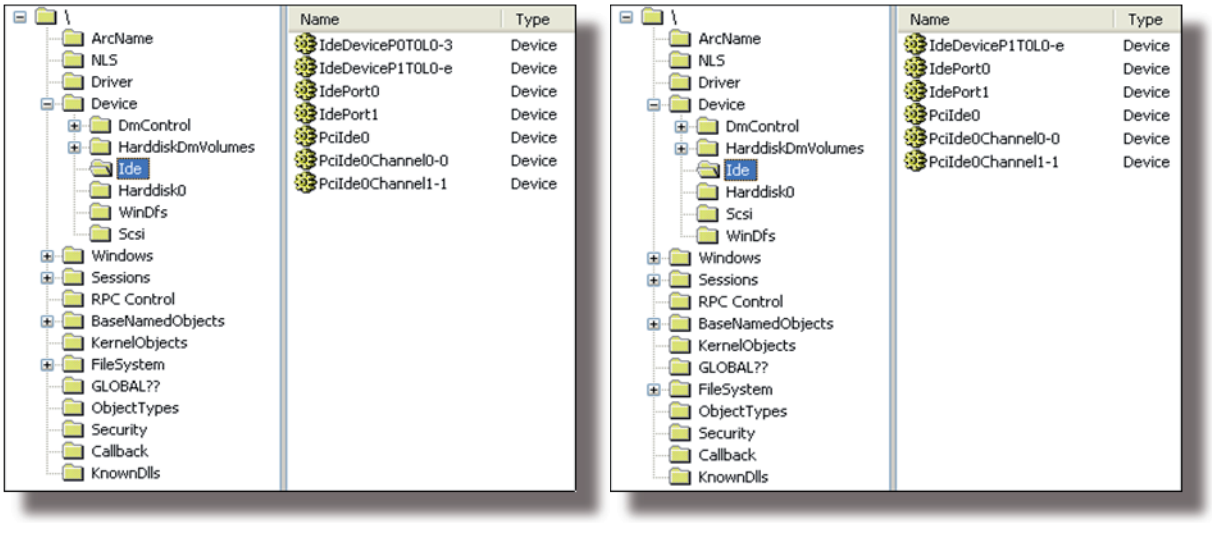


Figure 6: Clean system (left) and infected system (right) with the device 'missing'.

```
}
else if( a request to the atapi or oep resource
section, chksum, security data dir entry)
{
Replace the completion function of the current stack
element with its own function
}
}
```

So it is the completion function where the data is replaced.

When the first versions of TDL3 were found in the wild, some developers of anti-rootkit software made corresponding changes to their products so that they would at least detect the rootkit. However, virus writers were quick to respond and created new versions of the malware featuring new interception techniques which are harder to detect.

The dispatch table of the compromised driver remains clean. The authors of the rootkit used a non-standard approach. They simply 'stole' from the atapi driver the device object that is working with the system drive they intend to use (see Figure 6).

The abnormality can only be detected with a debugger (see Figure 7) – an unknown device using an unknown driver. Moreover, the DRIVER_OBJECT header of the 'unknown driver' is corrupt while the driver is removed from the system drivers list (as well as the 'stolen device'). The driver object is created by the rootkit to hide sectors of the hard drive and provide the malware with access to the hidden sectors. It has already become visible, but you still need to find or guess a device with a name comprised of eight random characters.

```
kd> !object \Device\Harddisk0\dr0
Object: 8179b030  Type: (817baad0) Device
    ObjectHeader: 8179b018 (old version)
    HandleCount: 0  PointerCount: 3
    Directory Object: e1342378  Name: DR0
kd> !devstack 8179b030
  !DevObj    !DrvObj            !DevExt   ObjectName
  8179be08   \Driver\PartMgr    8179bec0
> 8179b030   \Driver\Disk       8179b0e8  DR0
  817933f0 814a35f0: is not a driver object
817934a8
!DevNode 8179fee8 :
```

Figure 7: Detecting the abnormality with WinDbg.

Developers of anti-rootkits will have to devise a new way to use a specified device object to find a real driver used by the device.

The debug output of the rootkit upon its launch is also quite unusual. It reveals that the virus writers have a passion for cartoons. For instance, it may display one of the following lines:

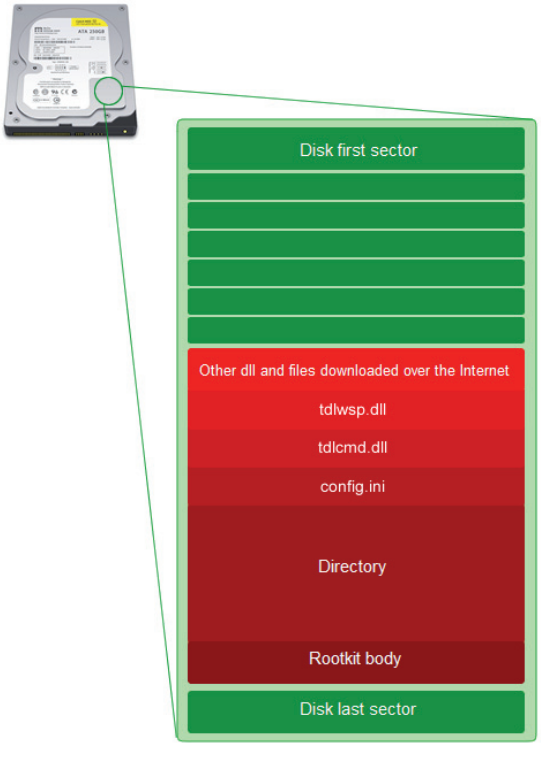
- Spider-Pig, Spider-Pig, does whatever a Spider-Pig does. Can he swing, from a web? No he can't, he's a pig. Look out! He is a Spider-Pig!
- This is your life, and it's ending one minute at a time
- The things you own end up owning you
- You are not your f*cking khakis

And in the later versions:

- Alright Brain, you don't like me, and I don't like you. But lets just do this, and I can get back to killing you with beer
- I'm normally not a praying man, but if you're up there, please save me Superman.
- Dude, meet me in Montana XX00, Jesus (H. Christ)
- Jebus where are you? Homer calls Jebus!
- TDL3 is not a new TDSS!

THE ROOTKIT FILE SYSTEM

At the end of the hard drive the rootkit occupies a certain area in which it stores its body and the virtual drive. The structure of a physical drive in a compromised system looks like this:



Sector numbers of the virtual drive increase from the upper sectors to the lower ones and the rootkit uses the negative offset starting from the sector that is used as a descriptor of the virtual directory (Figure 8). So, expanding backwards, it can overwrite data in other sectors of the physical drive.

File metadata and other information is placed in one file in the hidden disk drive. The size of the metadata is 12 bytes and it has the following format:

```
+00 Signature [TDLD – a directory, TDLF – a file, TDLN
– a file from the Internet]

+04 an ordinal number of a sector with valid data

+08 data size, if the sector provides sufficient space
for storage or if zero is not set for the preceding
field, the offset from file data to the next sector
where the file code is stored (i.e. +0xC for metadata,
so the field usually contains 0x3F4, 0x3F4 + 0xC =
0x400)
```

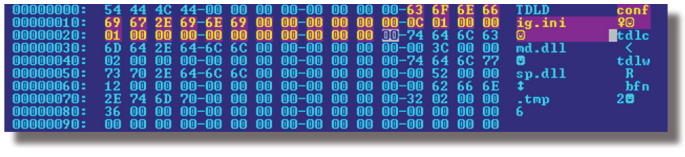


Figure 8: BackDoor.Tdss.565 virtual directory descriptor.

In Figure 8 you can see three files written onto the disk during the rootkit's installation (config.ini, tdlcmd.dll and tdlwsp.dll) and the bfn.tmp temporary file downloaded from the Internet. All sectors locating the drive are encrypted using RC4. The same encryption algorithm is used by other components that are not involved in the operation of the file system. The file described above is encrypted using the bot ID stored in config.ini. After decryption it appears as a set of commands for the rootkit (Figure 9).

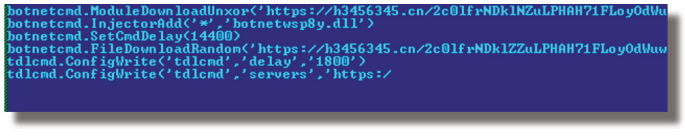


Figure 9: Contents of bfn.tmp.

Figure 10 shows a descriptor for the BackDoor.Tdss.1030 directory. Here we can see new file metadata fields and data for separate files of the rootkit body (tdl) and original resources of the infected file (rsrc.dat).

The directory incorporates a metadata structure and subsequent file entries. The size of each entry is 32 bytes (Figure 11 – an entry on Figure 7 is highlighted).

The first 12 bytes of the file descriptor contain metadata with the TDLF or TDLN signature, the number of the next sector and size placed at the beginning. For example, in Figure 12 you can see the specified file size 0x10C bytes.

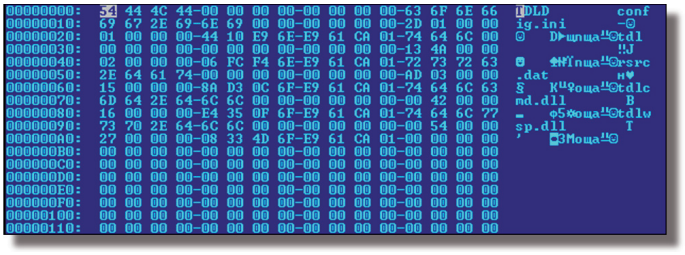


Figure 10: BackDoor.Tdss.1030 virtual directory descriptor.

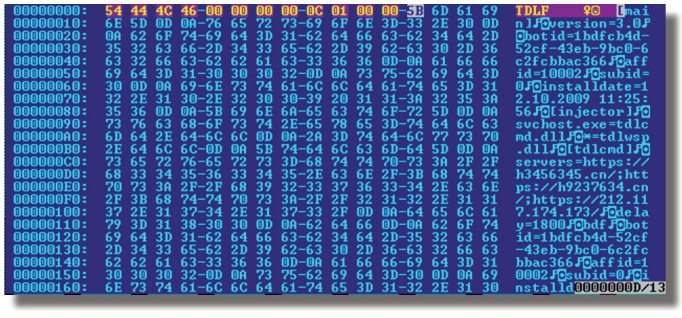


Figure 11: File descriptor.

In the rootkit's file system, a sector containing data is followed by a 'trash' sector since the rootkit works with 0x400 byte units (Figure 12) instead of 0x200 (for standard systems).

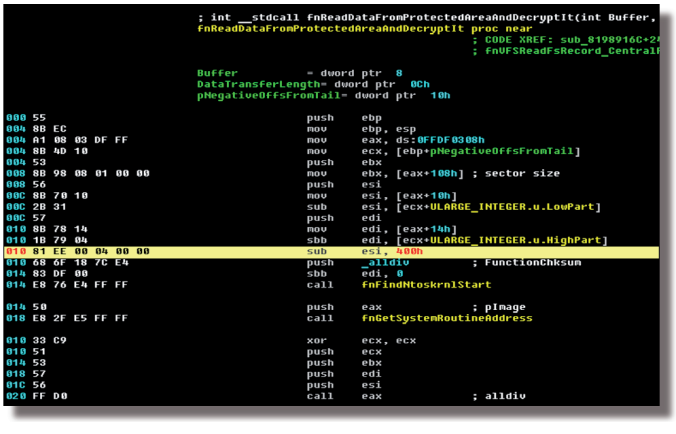


Figure 12: Reading sectors of the virtual drive.

CONCLUSION

All in all, new BackDoor.Tdss rootkits are sophisticated pieces of malware. Their detection and neutralization pose a serious challenge for anti-virus vendors – and, as has already been seen with BackDoor.MaosBoot (Mebroot), Win32.Ntldrbot (Rustock.C) and others, not all vendors are able to rise to that challenge.

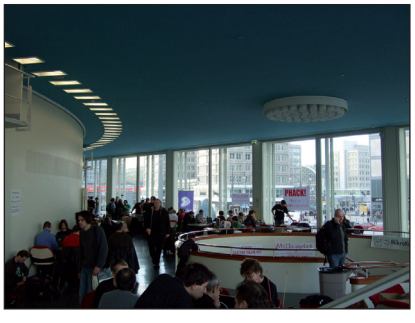
CONFERENCE REPORT

THE 26C3 CONGRESS OF THE CHAOS COMPUTER CLUB

Morton Swimmer
Trend Micro, USA

The Chaos Computer Club has officially been in existence since 1981 and has been organizing conferences ('congresses') since 1984. Through generations of leadership, a surprising number of themes have remained constant – for instance, the opposition to restrictions on the use of technology as well as technology's role in society. These themes play a large role in the Congresses, although there is also plenty of space and time dedicated to 'traditional' hacking and LAN partying. In the end, like any good conference, it is an information fest.

HERE BE DRAGONS



The 26th Congress was held over four days between Christmas and New Year in Berlin, Germany, with the slogan 'Here be Dragons'. By the first day, the Congress was so full of dragons that only a few tickets remained for day visitors and those travelling from afar. In all, it was estimated that 4,230 people participated on site. Oversubscription had been anticipated since the same problem had occurred last year, and an attempt was made to offer offsite participation through video streaming to remote chapters and anyone else who was interested. This was also useful for people who, like me, were on site, but could not get into the lecture halls due to overcrowding.

Unlike at the last event, there were no big disclosures this time. Instead, we heard about many incremental developments that are still very significant. With topics ranging from the politics of information to quantum cryptography, it is not possible to cover the hundreds of hours of material presented, so I will focus on a few select topics.

In the area that I roughly describe as information politics, there were quite a few presentations concerning events in Germany, such as recent attempts at Internet censorship and the country's data retention laws. There was also a presentation by the *Wikileaks* people about their activities and the negotiations going on with the government of Iceland to create an information free haven. While *Wikileaks* has its own agenda in needing such a free haven, this would also provide a location for depositing illicit material that some people in the security community would rather keep offline – so it will be interesting to see what form, if any, this takes.

One trend that continued this year was hardware hacking. My favourite hardware projects are those that follow the open source principle down to the hardware level to produce working quadrocopters that can easily be built from scratch using all the documentation that is available online. Two such projects were on site this year, Mikrokopter.de and Ng.uavp.ch, and both provide an open development process as well as a shop for buying parts if you don't want to make your own circuit boards, etc.

The idea of augmenting open source development with commercial interests was also manifest in another hardware project, the Makerbot, a 3D printer. Similarly, the Blinkenlights project and others were back again with various kits one could build in the hardware room in the basement. This was the first Congress to have a dedicated hardware room containing piles of soldering irons and other kit – which demonstrates the prominence of hardware hacking today.

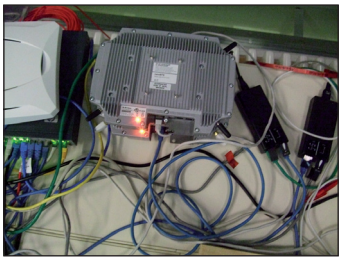
It wasn't a big surprise, then, that many of the hacking talks were about hardware. Continuing a theme from last year's Congress, there was a talk about the lack of security in the Swiss *Legic Prime* RFID cards which, while already deprecated, are still being used in physical access control – even at some airports. This analysis was performed by the same group who analysed the *Mifare Classic* cards last year. Needless to say, they found *Legic Prime* to be very lacking in security, to the point that they could emulate master cards granting access to all readers in a group. This should worry more than *Legic*'s customers and also demonstrates why security through obscurity is not a long-term strategy.

I CAN HEAR YOU

The big topic of the event was GSM technology, the largest phone network in the world. It has been known for over a decade that the GSM A5/1 cipher is broken. Initially, this had no practical value, but over time more evidence has emerged of successful practical attacks, though details have never been fully disclosed. Now, a group that includes Karsten Nohl and Chris Paget has explored enough of GSM and A5/1's weaknesses to mount a concerted attack against it. They have created an optimized variant of rainbow tables for this purpose and have started a distributed effort to generate them.

Effectively, this means that GSM privacy is dead if an attacker gets enough packets with known plaintext – which is highly likely, though there are some complications. GSM uses frequency hopping as well as time slicing below the encryption. The former means that you need to be able to predict the frequency on which the next packet will be transmitted, and this requires knowledge of the key. The way around this problem is to use two slightly modified universal software radios (USRP) to capture all traffic until the known plaintext attack has found the key, at which point they can lock onto the frequency hopping directly. More modifications need to be made to the USRP to make the attack more practical and offload much of the work now done by computer to the FPGA in the USRP, but the attack is feasible.

With the A5/1 algorithm now practically broken, what alternatives are there? Unfortunately, the newer GSM cipher, A5/3, is also theoretically broken. So far, there is no practical attack, so GSM operators could buy themselves some time by moving to this insecure algorithm, but they should not be under any illusions that doing so would 'solve' the privacy problem. It also won't solve the problem that the phone always trusts the base station, so rogue networks can easily be set up to disable encryption and capture all data. A phone could be made to detect attacks like this, though so far none are known to implement any phone-side security. It was clear from the talk that there are some deep design bugs in GSM that cannot easily be mitigated. There is a good chance that GSM security directly affects GPRS and EDGE data traffic, but any consequences for UMTS are so far unknown.



The talk was accompanied by others about GSM security and other activities in this field. There was a room dedicated to GSM hacking from which a private GSM phone network was being run. As permission had been obtained to run this experimental network, it couldn't be classified as 'rogue', but was an indication of what can be done. A lot of the activity in that room revolved around looking at various aspects of GSM phone technology, which meant a lot of hardware hacking.

Other GSM talks covered topics including the fuzzing of the phone-side operating system from either the PDA-side OS or the base station, or fuzzing the base station (and therefore the cell) from the phone. These activities are limited by the fact that there are very few vendors of GSM RF chips and it is hard to get at the documentation for them. Fuzzing is one way of finding out more despite the lack of documentation. Documentation projects have evolved around the *Nokia DCT3* series phones and the *TI TSM320* chips. I was told that the phone-side operating system of the *OpenMoko* phones is currently being reverse engineered (the PDA-side operating system is already open source).

There is increasing use of femtocells to fill GSM coverage gaps by routing phone traffic from the small GSM transceiver base stations to the telecom via the Internet. Philippe Langlois talked about the frequent lack of proper IPSec security of these devices and how one can access the SS7 or SIP signalling data. Femtocells are also an alternative to the micro cell base stations used in previous attacks and may make it even easier to set up rogue networks. I expect to hear more about these devices in future.

QUANTUM LEAP

Another impressive talk was on breaking quantum key exchange, which has been proven to be secure. Just as traditional cryptography can be vulnerable due to faulty implementations, it turns out that photon emitters and receivers are prone to attack, making a man-in-the-middle attack feasible. Qin Liu and Sebastien Sauge of the Quantum Hacking group at the Norwegian technical and scientific university, NTNU, had previously developed this attack on their campus and have now created a flight-case with which they can take a demonstration on the road.

Moving onto network security, the IPv4 address space is becoming a scarce resource and competition for address blocks is intensifying. At the same time it is not unknown for a block to become orphaned as companies go bankrupt or forget to track their assets. 'Nibbler' described how he was able to regain four address blocks despite not actually being the owner in the strictest sense (although they had been under his legitimate control at some point) by persuading RIPE to release the ASNs to him. While the Internet Assigned Numbers Authority (IANA) sets strict policies, the regional Internet registries are often lax in enforcing them and very old IP address spaces often don't fall under the more recent policies anyway. Nibbler went on a hunt for address blocks that might have been hijacked and believes he found one large space where the ownership has mutated over time in suspicious ways. While it seems unlikely that active address spaces would as easily fall prey to such persuasion, it is still a risk for companies who are slack with managing their Internet assets.

Fabian Yamaguchi of *Recurity Labs* talked us through a very convoluted attack involving various networking layers. It included the abusing of some known vulnerabilities and discovery of many more, but the most impressive aspect

was the weaving of these various unrelated and small vulnerabilities into a larger, effective attack. It shows that to withstand a determined attack, no vulnerability can be ignored even if individually the risk factor is low.

CERTIFIABLY INSECURE

Finally, Dan Kaminsky made an appearance at the Congress, this time talking about the X.509 certificate process. SSL/TLS, which is based on X.509 certificates, has held up surprisingly well, but cracks were already beginning to show prior to last year's attack against the scheme whereby a group was able to engineer a root certificate based on MD5 using the known vulnerability of that hash algorithm. Dan talked us through various other weaknesses in both the X.509 certificates and the general certification process. The economics of the process means there is effectively a race to the bottom as far as security is concerned. There are problems with the X.509 delegation approach which leads companies either to (nearly worthlessly) self-sign certificates to avoid constantly needing to purchase new ones from the certificate authorities, or to purchase the right to become a signing authority. There are also technical problems with delegation and the way that various implementations interpret them. Furthermore, MD2, the even less secure grand-daddy of MD5, is still in use – though all major browsers now have shunned its use and will probably remove support for that algorithm soon.

Dan's proposal is to eventually abandon X.509 as a public key infrastructure in favour of DNSSEC once its root key has been signed. His arguments are compelling, but it remains to be seen whether the major vendors will implement DNSSEC authentication. Most users don't understand enough about crypto to demand it, and moving to a new infrastructure is an upheaval for the vendors. However, enterprise customers – who have more clout with the vendors – should be particularly interested in the flexibility and security of a DNSSEC-based PKI, so we may see it rolled out sooner than we think.

NO DRAGONS LEFT BEHIND

Nearly all of the presentations are available online at http://events.ccc.de/congress/2009/, and many of the slides and other pieces of information can be found there too. I haven't been able to cover things like BIOS hacking, user-space virtualization, port scanning or web application fingerprinting in this article, but the relevant papers are all available online. It may not be the same as being there in person, but in future it may be impossible to get in without lining up a day in advance anyway!

TUTORIAL

MEMORY ANALYSIS – EXAMPLES

Ken Dunham
iSIGHT Partners, USA

In last month's introduction to memory analysis (see *VB*, February 2010, p.15), three distinct phases of operation were identified: analysis of a live system (triage), dumping of volatile data to a file (capture), and analysis of combined data (analysis). This follow-up article walks through the whole process using Haxdoor as an example.

INTRODUCTION TO HAXDOOR

It is helpful when learning new tools and techniques to start with a known sample. The variant of Haxdoor used in this demonstration has an MD5 value of 9bb6fbb9dfaff 0467d329284892d4e55. It uses kernel-level rootkit tactics to conceal processes, files and registry changes. Haxdoor is a well-known malware family due in part to its use in a phishing campaign targeting the Swedish *Nordea* bank [1]. With seven to eight million Swedish kronor having been siphoned away by the attackers, *McAfee* called this incident the 'biggest ever' online bank heist at the time of disclosure.

Haxdoor can be sent to a victim through numerous vectors, including email, web exploitation and more. Once executed, this particular variant creates the following files, many of which are hidden from *Windows*:

- kgctini.dat
- lps.dat
- qo.dll
- qo.sys
- svjvpn.sys
- svjvpn.dll
- svkvpn.sys
- Temp/W01083060Z (directory)

It also makes the following *Windows* registry changes:

```
HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\
Winlogon\Notify\svkvpn
Startup = "ER03Sb5fex"

HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\
Winlogon\Notify\svkvpn
DllName = "svkvpn.dll"

HKLM\SYSTEM\ControlSet001\Control\SafeBoot\Minimal\
svjvpn.sys
(Default) = "Driver"

HKLM\SYSTEM\ControlSet001\Control\SafeBoot\Network\
svjvpn.sys
(Default) = "Driver"

HKLM\SYSTEM\ControlSet001\Enum\Root\LEGACY_
SVJVPN\0000\Control
```

```
ActiveService = "svjvpn"
HKLM\SYSTEM\ControlSet001\Enum\Root\LEGACY_
SVJVPN\0000
Service = "svjvpn"
HKLM\SYSTEM\CurrentControlSet\Services\svjvpn\
```

In addition, it injects explorer.exe to run hidden in memory and it may attempt communications with skyinet.info.

TRIAGE

After infecting a test *Windows* operating system with Haxdoor, triage begins.

1. Windows Task Manager (CTRL-ALT-DELETE)

The first step is to look at the *Windows Task Manager*, sort by image name and look for any processes that are missing from the list, and any processes on the list that should not be there. Having done this, we can see that explorer.exe is missing from the list – it should be visible. This is an indication that the process has been injected and hidden by a rootkit (Figure 1).

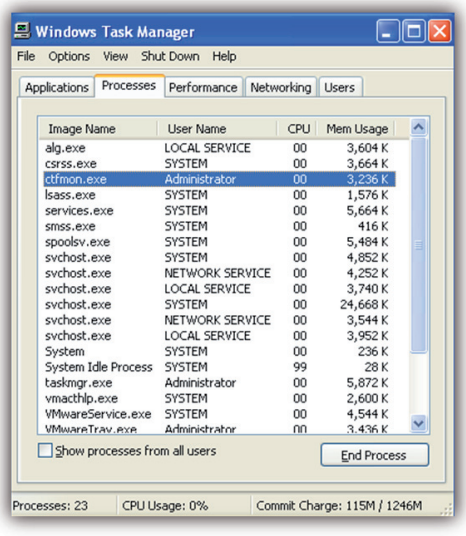


Figure 1: Explorer.exe is not visible in Windows Task Manager. This is an indication that it has been injected and hidden by a rootkit.

2. Process Explorer

The next step is to look for extra and/or missing processes in *Process Explorer* – some malicious programs are visible using this tool but not with *Windows Task Manager*, indicating a possible rootkit process. However, in this case *Process Explorer 11.x* does not reveal any new information.

3. FPort

The next step is to run *FPort* and dump the results to a file. *FPort* reveals the following output:

```
FPort v2.0 - TCP/IP Process to Port Mapper
Copyright 2000 by Foundstone, Inc.
http://www.foundstone.com

Pid   Process    Port      Proto Path
932   svchost    -> 135    TCP   C:\WINDOWS\system32\
                                 svchost.exe
4     System     -> 139    TCP
4     System     -> 445    TCP
684   alg        -> 1028   TCP   C:\WINDOWS\System32\
                                 alg.exe
1588  Explorer   -> 16016  TCP   C:\WINDOWS\Explorer.EXE
1588  Explorer   -> 16661  TCP   C:\WINDOWS\Explorer.EXE
1588  Explorer   -> 43818  TCP   C:\WINDOWS\Explorer.EXE
1588  Explorer   -> 47762  TCP   C:\WINDOWS\Explorer.EXE
1588  Explorer   -> 123    UDP   C:\WINDOWS\Explorer.EXE
684   alg        -> 123    UDP   C:\WINDOWS\System32\
                                 alg.exe
4     System     -> 137    UDP
0     System     -> 138    UDP
932   svchost    -> 445    UDP   C:\WINDOWS\system32\
                                 svchost.exe
4     System     -> 500    UDP
1588  Explorer   -> 1025   UDP   C:\WINDOWS\Explorer.EXE
1588  Explorer   -> 1900   UDP   C:\WINDOWS\Explorer.EXE
0     System     -> 1900   UDP
1588  Explorer   -> 4500   UDP   C:\WINDOWS\Explorer.EXE
```

A quick analysis of the list above reveals explorer.exe (PID 1588) communicating on TCP ports 16016, 16661, 43818 and 47762 in addition to other traffic. Since explorer.exe was hidden before, probably injected, and is now found to be associated with ephemeral port activity, this is an area to focus on.

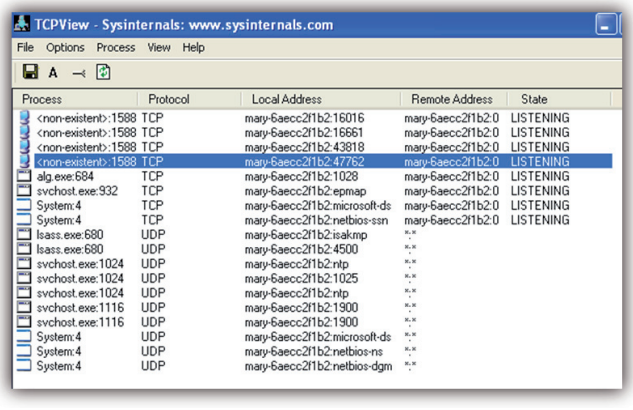


Figure 2: TCPView reveals the same activity as seen with FPort.

4. TCPView

TCPView is the next logical step, since it is a great tool for providing a quick visual overview of any running processes that are responsible for TCP communications. In this case *TCPView* produces the output shown in Figure 2, revealing 'non-existent' process names communicating on these same ports.

At this point we have confirmed an injected process and identified two TCP ports of interest. We have yet to identify file and registry changes. *IceSword* is the next tool we will use to analyse the system and focus on these initial leads.

5. IceSword

IceSword highlights any data it believes to be associated with rootkit activity. Screenshots in Figures 3–12 show snippets of clues and proven rootkit functionality via this tool.

Figure 3: IceSword highlights (in red) a malicious rootkit process injected into explorer.exe.

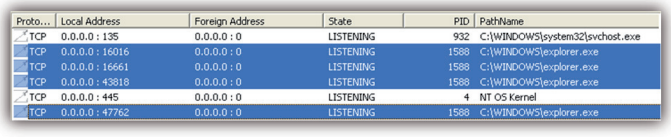


Figure 4: IceSword port analysis reveals the same ports seen with FPort and TCPView.

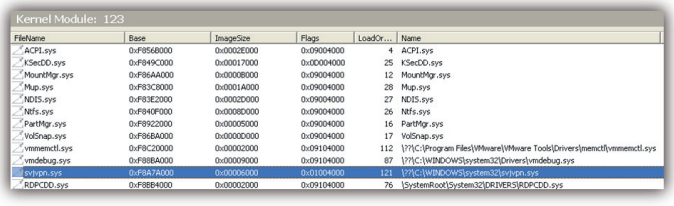


Figure 5: IceSword reveals the kernel-level rootkit and its location.

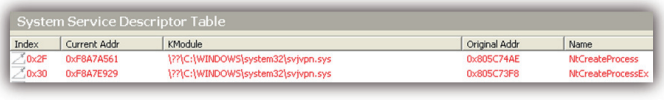


Figure 6: IceSword SSDT reveals rootkit activity.

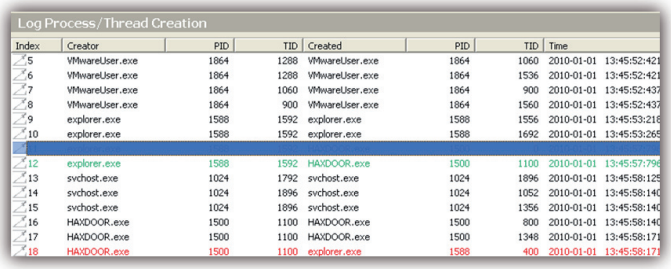


Figure 7: IceSword logs show Haxdoor injecting explorer.exe.

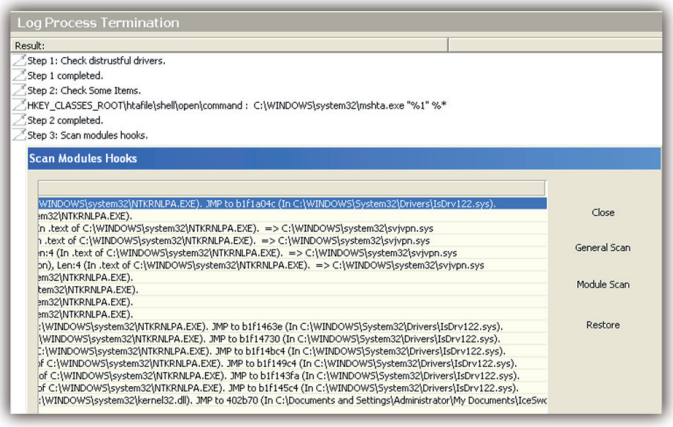


Figure 8: IceSword scans reveal several hooks.

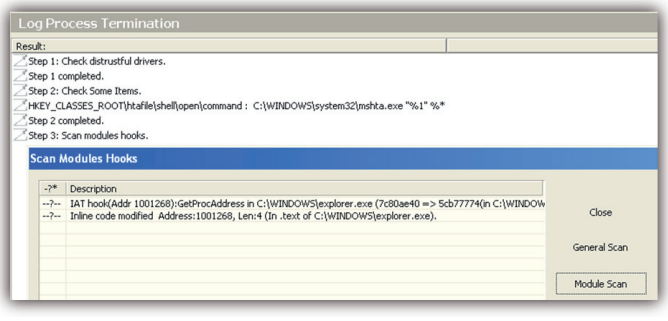


Figure 9: IceSword is able to scan specific processes too, like PID 1588 explorer.exe in this case.

CAPTURE

With triage completed, the next step is to capture physical memory to a file for further analysis.

The open source *MDD* tool was used in the capture of RAM to a file on the infected system. *IceSword* and a tool from

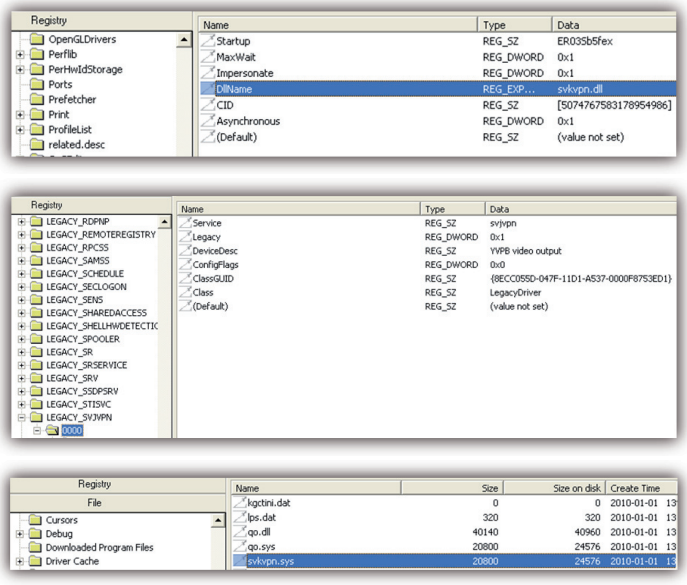


Figure 10: IceSword shows registry changes.

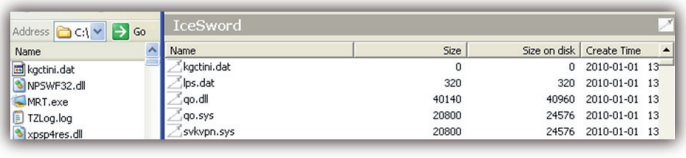


Figure 11: IceSword shows all the files hidden by the rootkit.

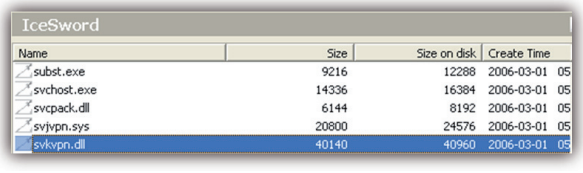


Figure 12: Be careful looking only for newly created files. The DLL has a modified MAC time.

GMER called *catchme.exe* were used to capture rootkit files. Once captured, the memory image can be analysed with the *Volatility Framework* and files are captured via standard behavioural analysis, multiscanners, sandboxes, MD5 and file data open source queries, and more.

The MD5 of the main Haxdoor executable has already been analysed, with public data available at the following locations:

- http://www.threatexpert.com/report.aspx?md5=9bb6fbb9dfaff0467d329284892d4e55
- http://www.sans.org/reading_room/whitepapers/honors/stealth_for_survival_threat_of_the_unknown_176

This type of data aids the researcher in identifying what other analysis and/or incident data exists for the sample and may prompt the researcher to revisit the infected computer for additional files or behavioural tests and/or identify where to focus the *Volatility Framework* analysis.

ANALYSIS

Before beginning *Volatility Framework* analysis we need a solid reference from what is known or suspected about the code being investigated. This aids in the specific commands and exports performed within the *Volatility Framework* for the memory dump analysed.

The following information is known about this specific dump:

- It creates several files: kgctini.dat, lps.dat, qo.dll, qo.sys, svjvpn.sys, svjvpn.dll, svkvpn.sys and Temp/W01083060Z (directory).
- It makes changes to the *Windows* registry referencing svkvpn and svjvpn.sys.
- It injects explorer.exe PID 1588 and may attempt communications with skyinet.info.

Of course, the initial investigation may not have turned up all hidden components on a system. As such, the list above is only an initial triage in confirming and capturing data related to the attack. Additional measures may be required, such as dumping all processes to files and inspecting them for possible hostile content.

In our example, the output from the DLLlist and Files commands confirms the injected rootkit DLL in explorer.exe (PID 1588):

DLLlist

```
explorer.exe pid: 1588
Command line : C:\WINDOWS\Explorer.EXE
Service Pack 3
Base        Size      Path
0x1000000   0xff000   C:\WINDOWS\Explorer.EXE
...
0x2c90000   0x52000   C:\WINDOWS\system32\svkvpn.dll
```

Files

```
Pid: 1588
…
File    \WINDOWS\WinSxS\x86_Microsoft.Windows.Common-
Controls_6595b64144ccf1df_6.0.2600.5512_x-ww_35d4ce83
File    \WINDOWS\system32\svkvpn.dll
```

A socket scan reveals specific offsets of interest related to the injected port activity:

```
PID   Port   Proto Create Time                Offset
---   ----   ----- ------------------------   ----------
1588  16016  6     Fri Jan 01 17:18:44 2010   0x020d80e8
1588  16661  6     Fri Jan 01 17:18:44 2010   0x024cc600
1588  16016  6     Fri Jan 01 17:18:44 2010   0x11e5e0e8
1588  16016  6     Fri Jan 01 17:18:44 2010   0x16d560e8
1588  16661  6     Fri Jan 01 17:18:44 2010   0x17752600
```

Finally, a module scan reveals the SYS file details of interest:

```
File: \??\C:\WINDOWS\system32\svjvpn.sys
Base: 0x00f8a42000
Size: 0x006000
Name: svjvpn.sys
```

CHALLENGE

Another rootkit has been run in *Windows*. The reader is invited to determine what family of code it belongs to and what is malicious based on the brief description below; the author can be contacted for the answer.

When run within *VMware*, the code runs in memory and then disappears from *Process Explorer 11.x*. Yet when *TCPView* is run, it terminates immediately instead of running like it should, and *Windows Task Manager* won't run in memory. *FPort* still works and reveals svchost activity over TCP port 58318:

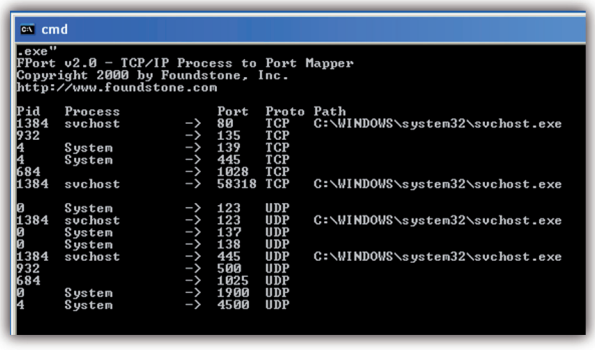


Figure 13: Svchost reveals TCP activity likely related to a rootkit in memory.

IceSword reveals the same TCP port activity for svchost.exe (PID 1384). It also reveals three hidden files in the *Windows* System32 directory: msux, msad32.dll and msur.exe. A search for the DLL in *Windows* Regedit reveals changes to the registry in ShellServiceObjectDelayLoad (Figure 14).

A DLLlist analysis with the *Volatility Framework* confirms that the DLL in question is injected into the svchost PID 1384:

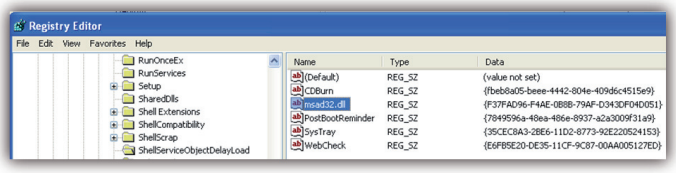


Figure 14: Changes to the Windows registry are clearly visible with Regedit.

```
svchost.exe pid: 1384
Command line : svchost.exe
Service Pack 3
Base        Size     Path
0x1000000   0x6000   C:\WINDOWS\system32\svchost.exe
0x10000000  0x46000  C:\WINDOWS\system32\msad32.dll
0x7c900000  0xb2000  C:\WINDOWS\system32\ntdll.dll
```

A files analysis with the *Volatility Framework* provides the following details for PID 1384:

```
Pid: 1384
File  \Documents and Settings\Administrator
File  \WINDOWS\WinSxS\x86_Microsoft.Windows.Common-
Controls_6595b64144ccf1df_6.0.2600.5512_x-ww_35d4ce83
File  \WINDOWS\WinSxS\x86_Microsoft.Windows.Common-
Controls_6595b64144ccf1df_6.0.2600.5512_x-ww_35d4ce83
File  \WINDOWS\WinSxS\x86_Microsoft.Windows.Common-
Controls_6595b64144ccf1df_6.0.2600.5512_x-ww_35d4ce83
File  \Endpoint
File  \Documents and Settings\Administrator\Local
Settings\Temporary Internet Files\Content.IE5\
index.dat
File  \Documents and Settings\Administrator\Cookies\
index.dat
File  \Documents and Settings\Administrator\Local
Settings\History\History.IE5\index.dat
File  \WINDOWS\WinSxS\x86_Microsoft.Windows.Common-
Controls_6595b64144ccf1df_6.0.2600.5512_x-ww_35d4ce83
File  \WINDOWS
File  \
File  \ROUTER
File  \ROUTER
File  \Endpoint
File  \WINDOWS\system32\raschap.dll
File  \WINDOWS\system32\logonui.exe
```

When run a second time on a clean *Windows XP Professional* system, msrf32.dll is created in the *Windows* System32 directory with similar size, behaviour and files.

For the answer to this challenge, contact the author at kend@kendunham.org.

REFERENCE

[1] http://news.zdnet.co.uk/security/0,1000000189,39285547,00.htm.

PRODUCT REVIEW

CA INTERNET SECURITY SUITE PLUS 2010

John Hawes

When we first looked at *CA*'s new product for the home market (in the recent VB100 comparative on *Windows 7* – see *VB*, December 2009, p.16), we were surprised and somewhat baffled by its radical new approach to providing its users with control and information. However, feeling that our first response – based as it was on the brief period spent using the few small areas of the interface required for the VB100 test – might have done the product little justice, it seemed appropriate to give it a second chance.

CA was kind enough to ship over a full boxed version of the product, which enabled us to evaluate the full user experience from store shelf onwards. Taking the funky-looking box into the lab, we put it in a prominent place until we had time to slip the glossy CD into a test machine and look at what was on offer.

COMPANY AND ONLINE PRESENCE

Formerly known as *Computer Associates*, *CA* is one of the great survivors of the software industry, boasting a length of service and endurance matched by only a few fellow giants. The company went public almost 30 years ago, and existed for many years before that. Between then and now (with 13,000 employees and revenues of over $4.2 billion, according to 2009 figures) it has absorbed an impressive roster of more than 50 other firms; the acquisition of yet another – cloud services firm *3Tera* – was announced while putting this review together.

The list of acquisitions includes several firms that have contributed significantly to *CA*'s security offerings. These include *Pest Patrol*, which formed the core of the company's anti-spyware offering and was acquired in 2004. Looking further back we find *Cheyenne*, an early participant in VB100 comparative reviews, whose *Inoculan* product evolved into *CA*'s *InoculateIT*. The other major string to the company's anti-malware bow, again appearing as an independent entity in early VB100 reviews, was the *VET* engine, purchased in 1999, which eventually came to supplant the *InoculateIT* line after operating alongside it.

More recently, the process of bringing technology and expertise in-house through the acquisition of established firms has been reversed somewhat, with much of the burden of developing *CA*'s anti-malware components now farmed out to mammoth outsourcing firm *HCL*. This move has sparked a series of changes in the product range, of which the facelift for the consumer line is the most recent and the most radical.

With such a sizeable company and such a diverse range of products and solutions available (the company's website lists 10 other high-level product categories alongside its security offerings), www.ca.com is an enormous and multi-faceted place. Initial investigations led only to information on the corporate product range, but eventually a home-user area was turned up in the online shop section. The home-user product line includes simple anti-virus as well as the suite, plus some 'PC Tune-up' solutions. Information was provided on all of these, the various functions and modules offered, awards and certifications received, and so on. A link is also provided for potential affiliates, with website owners offered handsome rewards for leading new customers to the *CA* site. *CA* itself is affiliated with the *Yahoo!* web Goliath, its solutions are given away to some *Yahoo!* users and, by way of return, *CA* users are offered *Yahoo!* toolbars – more on which later.

Delving even further into the *CA* website, we eventually found a less product-oriented security area at http://www.ca.com/us/global-technology-security.aspx. This 'Global Security Advisor' section provides all the usual data found on most security company websites: lists of the latest and most prevalent threats, alert meters, malware databases, a sample submission system, a glossary, research papers, articles, news pieces, webcasts, and of course a blog. There is also a forum, which seems fairly well populated but the section titles may be rather obscure to the uninitiated and the bulk of the traffic is clearly focused on enterprise issues.

Full support sections are provided for all the products. For the *Internet Security Suite* this is accessed via the shop and is hosted at cainternetsecurity.net, which seems to have yet more resources related to the product, including knowledgebases, guides, how-tos and yet more blogging, as well as a system for submitting support tickets. Having found more than enough information and tools, it was time to break the seal on the box and have a look at what was inside.

INSTALLATION AND CONFIGURATION

Breaking into the packaging proved a less simple operation than one might expect; the funky design of the new box avoids the traditional simplicity of rectangles and square corners, instead including a feature corner at a rakish angle to the other parts and a frustratingly difficult process of getting at the CD and licence code inside.

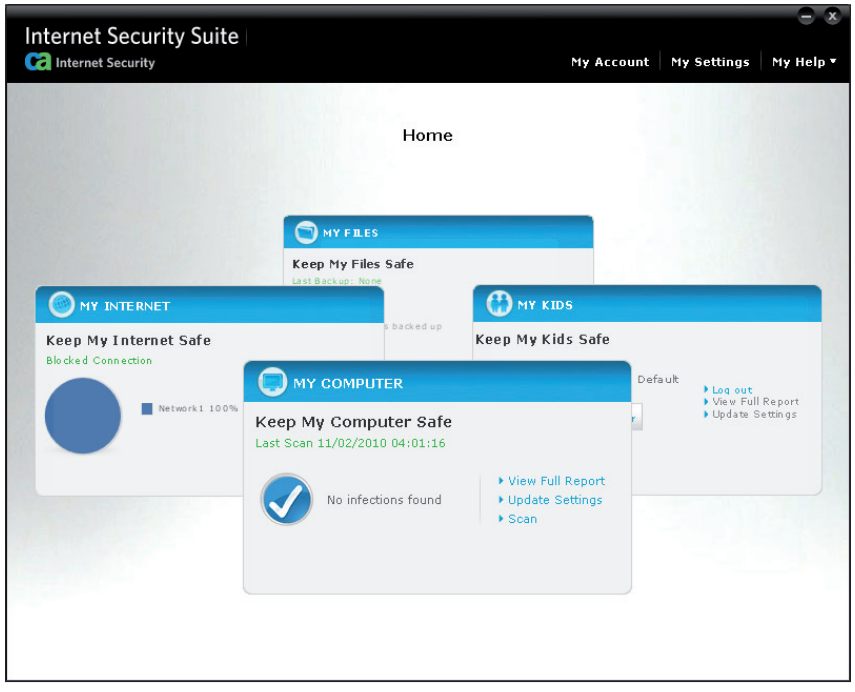
With this achieved (and the box reduced to little more than shreds), installing the product was something of an anti-climax in its straightforwardness, running through

all the standard steps, enlivened only by *CA*'s extremely thorough approach to EULAs. In this case the LGPL (Lesser General Public License) was included, as was an activation process involving both a lengthy product key and the set-up of an online user account. This allows the purchaser of the standard package to install the product on up to three systems simply by associating the installs with the same user account.

With the interactive stage over, the hands-off process of copying files etc. took a fair amount of time and included both a 'quick scan' of the target system and a reboot before it declared itself complete. With this done, we were surprised to find it still required an online update, having assumed this would have been part of the initial install and activation. There could have been a minor bug on a few of the test systems, as we noticed some went through several cycles of demanding updates even immediately after an update had completed; perhaps the product was simply being too thorough.

However, all these little niggles were barely noticeable once the new-look interface was up and running. An introductory walkthrough is offered, in the form of an animated, annotated guide, but anyone who's anything like us will want to start by playing around with the home page. In our previous, rather flippant summary, we described it as resembling the improbable systems used by the character Horatio Caine and his associates in the TV show *CSI: Miami*, with lots of sliding around of tabs in 3D. Rather than the traditional file-browser-inspired layout with tabs arranged down the left-hand side or along the top, here we have four 'panels' covering four major sections of the product, which rotate around when clicked to bring the selected one to the front. Clicking on the panel title opens the full window for that panel, while some main controls can be accessed from the smaller version. The main detail windows for each panel are further subdivided into tabs in a more traditional manner, with simple buttons along the top.

The 'My Computer' panel loosely covers the main malware detection functionality. Opening the panel brings up the usual configuration options, including scheduling, which all appear to be present and correct and simple to operate. There are also tabs for 'reports' and 'history', with history being detailed logging and information along the lines of summary statistics and graphs. The other main panels operate in a similar fashion, with some configuration and some reporting; while focused on a given panel, a row of buttons along the bottom provide access to the other panels and to the funky 3D home screen. Everything is in bright colours, with large buttons, ticks and crosses – presumably designed to be clear and unthreatening but, at least for us, and I suspect for many other experienced users of security software, a little confusing thanks to its considerable departure from the style we have become accustomed to. Perhaps this is a Luddite view however, and more flexible users will find themselves adapting with greater ease to a different way of thinking and working.



THREAT DETECTION AND PROTECTION

Regardless of the response of different user groups to the innovations in layout, it cannot be denied that the provision of the protective elements at the core of the security suite has been considerably simplified here. We have seen many products recently which seem to try to provide an impression of increased protection simply by dividing the components into a large and bewildering selection of separate areas – with malware scanning, real-time protection, anti-spyware, anti-spam, firewalling, HIPS, behavioural monitoring, web filtering and messaging monitoring all treated as standalone layers of protection within a single multifunction suite. Commendably, *CA*'s redesign has taken the opposite approach. All of these core protective functions are contained within two umbrella sections of the control interface.

The first section is familiarly entitled 'My Computer', and includes the functions generally covered by anti-malware products. One of the main links in the initial panel is entitled 'update settings', which is perhaps an unfortunate choice of terminology given the association of updating with traditional anti-malware. Most users reading this would assume that the link would lead to the settings for the updating process; however, this is covered in a separate section (reached from a 'settings' link displayed prominently at the top of the product, where various global options such as alerting, proxy usage and update scheduling can be managed). The link in the 'My Computer' panel is apparently intended to be read not as the settings for the update, but rather as an update to the settings. It leads to the standard set of options, such as enabling on-access and email scanning, scanning different types of files, default responses to detections and scheduling of on-demand scans. The content is fairly standard and the control system pleasantly laid out, with a good level of clarity and a decent degree of fine-tuning made available. The section also provides reporting, including statistical overviews and graphs, and more detailed history of scanning and detection activity.

Having performed numerous tests of *CA*'s products in our VB100 comparatives in recent years, there seemed to be little further to analyse here; the company has a pretty solid record of certification, generally showing excellent scores in our traditional test sets, but lagging somewhat behind the field since the introduction of the additional trojan and RAP (Reactive And Proactive) test sets, which include a broader range of content and greater degree of freshness. *CA* has shown keen interest in diagnosing whatever problems may be indicated by this, be they problems with its sample processing procedures or issues with our test methodology which may have introduced some unintended bias. We fully expect to see some solid improvement in *CA*'s scores in these areas in the near future, representing improvements both in the protection provided by the products and in the accuracy of our testing methods.

Moving along from this fairly standard area, the second top-level panel in the spinning 3D home page contains firewall, anti-spam and intrusion prevention. This is labelled 'My Internet', and as in the 'My Computer' section provides some top-level links in the mini panel on the home page, in this case leading to reporting, settings (again labelled 'update settings'), and a simple option to disable the firewall component entirely.

Digging deeper into the full control system for this section, the main screen shows a brief summary of blocked remote access attempts, a drop-down with standard tasks, including again the option to disable the firewall, a lockdown mode blocking all network activity, a purge of caches, and access to the full history log. The most important and detailed part is, of course, the full settings area, which provides access to the reporting system, which can provide fine-grained summaries of the various types of protection offered, and the history, with detailed logging, alongside the configuration controls.

The initial firewall set-up area is pleasantly clear and simple – something that has become ever more important in recent years as users become more aware of the necessity of firewalls but more likely to kick against any attempt to bewilder them with the traditional, port-number and protocol-heavy configuration systems of older types of firewall. Here, users are offered the choice of either a home network with file sharing and full permissions to all trusted applications, or a public network with more secure rules (the lockdown mode appears here too), with some advanced options offering the choice to apply rules to various protocols. Everything is presented in simple language where possible and an effort has clearly been made to make things as accessible as possible to users who lack a deep understanding of networking terminology.

The second tab provides an option for more experienced (or simply braver) users to create their own bespoke network rules, and this area is of necessity considerably more complex. It is laid out in a clear and logical manner however, with as much explanatory text as possible and useful links to more detail in the help system on each page.

Next up is a browser protection component, which by default only provides some basic cookie watching but can be configured to block various types of cookies, and also includes a pop-up blocker and a script blocker. These can be fine tuned to block various kinds of potential risks and can also be set up on a per-domain basis. A schedulable cache cleaner is also included here, once again in a simple and lucid manner, allowing non-expert users a good degree of control without demanding too much research and investigation to make sense of things.

After that comes an identity protection module, which can be filled with various sensitive pieces of information such as credit card numbers or the names of family members, and then prevents the protected data being transferred to websites or via email. Trusted sites can be specified to minimize unwanted interruptions of known-good transactions, and here too everything is nicely laid out with ample explanation provided. Some cursory investigation showed that it was pretty effective at preventing data leaking out.

The final area is labelled 'Web protection' and is split into two main components, each of which only has basic controls in the main interface. Most of the controls for these are provided in the browser or mail client, depending on

what is appropriate to the specific module. The first part is a spam filter, which works only with *Microsoft*'s *Outlook* and *Outlook Express*; in the main GUI there are only options to enable or disable integration with these, but when one of the clients is opened for the first time after installing the product a brief set-up process points it to existing folders of ham and spam for training, and allows the creation of an address whitelist. A toolbar is then added to the client to allow future tweaking of settings and lists, and also includes a handy search facility. Our current anti-spam testing set-up is designed primarily to measure the efficacy of gateway-level filtering software, but we hope to adapt it to enable some analysis of client-level offerings in the near future; we will return to this product to investigate how well it performs when we can.

Phishing protection is the last part of the product, and again in the main GUI there are only options to enable or disable it and to select which clients it integrates with. The list of supported clients includes various browsers, mail clients and chat tools, the most popular of which are enabled by default with others needing explicit activation; *Microsoft Word* is a notable inclusion in the additional set, while *Google Chrome* and *Opera* are notable for their absence.

Other controls are provided within the protected client; the set-up provides clear information on the safety of links and sites through a system of coloured markings which indicate how much information is available on their legitimacy. The current page is marked in a toolbar, while links can be flagged when hovered over if desired. This second part seemed a little intrusive when set up to always show advice and keep it close to the mouse, but it can be relegated to any area of the screen and can also be set to only show when the control key is held down. It seemed to be fairly speedy in its responses and had a good level of accuracy, quickly identifying major legitimate sites and warning about many less reputable ones.

OTHER FEATURES

Of the four main components of the product, the remaining two may be seen as additional functions beyond the narrow scope of malware protection, although of course they remain under the general sphere of security. The first is the 'My Files' component – essentially a back-up system. This seems pretty straightforward: various folders and areas can be selected to be backed up and saved to a secure archive, which can either be on the local system or – the recommended option for proper security of back-ups – on an external removable drive. The default is to include the whole of 'My Documents', but the browse window to select additional areas is simplified to exclude symbolic areas such as My Documents, so selecting only a subset of the commonly used storage area is rendered a little tricky for most untutored users.

The back-up itself is actually considerably more complicated than a simple archive, creating an executable which includes its own interface for restoring or 'migrating' backed up items. One initial test, backing up a folder containing two files of 5KB or so each, took two minutes and created a back-up file of over 40MB. Trying to restore this to a different location had some rather bizarre side effects, including closing down a *Windows Explorer* instance and, strangest of all, reversing the order of all the task bar icons. Obviously this situation was not an intended application of the tool; the product help was less than plain on the subject of how it is meant to be used, listing the jobs which could be performed but not really explaining the purpose of the complexity introduced, and little time was available for a more thorough investigation. The panel also provides detailed history and reporting on back-ups made and restored.

The fourth top-level panel is entitled 'My Kids' and provides some parental controls. Once again, this starts off fairly simply, with the controls for setting up rulesets for individual children. Having entered a name, there is a fairly clear section gathering passwords and existing

by-age-group default settings to the new profile. Some nice simple screens then appear for setting allowed or blocked times for Internet use and blocking of types of online content; an advanced section on top of this allows the parent to include specific sites. The final stage allows specific applications to be blocked, in both peer-to-peer and IM categories. These can either be banned outright or on a per-product basis, and for IM behaviour there is an advanced section which can be used to block keywords and even specific contacts.

So far so simple, but we had a little confusion with the implementation of the user system, which seems not to be tied to *Windows* users but instead is controlled entirely from within the product; the parent sets the rules, then logs the system into a chosen user in the GUI. Having a password set for each user allows an older child with broader privileges to log themselves in to override the stricter settings of younger children, but to control the overall rules the admin or parent user must log back in. A pop-up appears to indicate which user has been switched to.

After a little playing around we soon figured out how it operated, and it does seem a sensible option for the average home user, who is unlikely to make proper use of the *Windows* User subsystem, instead simply leaving the default (usually admin) user logged in pretty much permanently. This is an interesting example of tailoring the user process to real-world usage rather than the 'proper' way of doing things. Again, detailed logging and reporting is included, and appears fairly clear and simple to use.

The last item to discuss is the help system, which comes in several forms, all hosted in a slick black interface blending nicely with the main GUI – a standard help file, a set of 'top solutions' and a series of instructional videos. The help file follows a by-button path, explaining what each section of the control system relates to, and offers some more detailed explanations than can be gleaned from reading the buttons and so on. However, it provides little by way of holistic explanations of the intent or purpose of a given component. These sections are all properly linked to from within the main interface, with each area providing a link to the matching explanation – something which far too many products fail to implement as thoroughly as is done here.

This more task-oriented approach is covered by the 'top solutions' section, which functions in the form of an FAQ and provides detailed steps to carry out a range of tasks and to solve some common problems. Oddly, these open not within the main help interface but in new instances of *Internet Explorer* (not the default browser of the system but specifically *IE*). The videos likewise pull up *IE* and some very slow video rendering from *Adobe*, which we gave up on after waiting for ten minutes of buffering; most of the clips offered by this 2010 product seemed to refer to IS2009. The help subsystem also includes links to the online community and forums. Overall, the system seems to provide a decent range and level of instructional matter, but suffers somewhat from a lack of joined-up design and logical implementation.

CONCLUSIONS

As we observe the ever-growing and ever-evolving range of security products available to computer users through our VB100 comparatives, interface design is something that has increasingly come to our attention as a significant differentiator between solutions. With more and more features included in modern suites, the user is required to spend more time interacting with products which have traditionally aimed for a 'set and forget' paradigm. The selection of these additional features can vary fairly widely between products, as each vendor combines different options from the pool of common choices, but most suites combine a core set of elements with possibly a few unusual and even unique extras. In the same way, interface design tends to conform to a basic standard format, with some quirks and oddities in each product; differences tend to be in quality of implementation rather than drastic departures from the accepted norm.

CA, along with its partner *HCL*, has taken a brave and unusual approach to its product design, and in its efforts to provide a more open and usable experience has had some success. While the set-up may be somewhat confusing at first to those users already well used to existing practices, it does feel that the new workflows presented here would be perfectly usable by those not so set in their ways. The modesty of the product is particularly notable, with the usual flooding of interfaces with separate sections to make a product appear more complete eschewed in favour of simplicity and elegance. The network protection component is a particularly clear example of this.

There is certainly room for improvement in a few areas, particularly as far as we are concerned in the RAP detection scores, but purely in terms of its design and implementation there is much to commend here. We have often noted in these reviews the importance of empowering users to take control of their own security by de-obfuscating the management of security solutions. This product takes an interesting and fairly successful step along the important path towards allowing normal people to understand what risks they take with their computers, and how to keep themselves safe from danger.

COMPARATIVE REVIEW

VBSPAM COMPARATIVE REVIEW

Martijn Grooten

Thanks to many programs that are freely available on the Internet, building a spam filter is not rocket science. However, building a *good* spam filter is not a trivial task. And, with the email security market still growing and new products appearing every month, many customers will wonder whether a hitherto unknown product, with a shiny website and an impressive sales story, is actually any good.

The purpose of VBSpam testing is to provide an easy-to-recognize certification that tells potential customers that a product does what a good spam filter should do: i.e. block the vast majority of spam, with very few false positives. As such, we are delighted that, for the first time, all of the products in this month's test achieved a VBSpam award. This does not mean that no bad products exist – after all we only test products that have been submitted by their developers – but it does demonstrate that there is plenty of choice for customers, as well as a healthy amount of competition for product developers.

But in spam filtering, the devil is in the details. With recent reports suggesting that up to 95% of email traffic is spam, email can only be a viable form of communication for businesses if the vast majority of that spam is blocked – and blocking one or two per cent more will have a huge impact on users' inboxes. Similarly, a very low false positive rate is essential, and even a couple fewer false positives every month will significantly improve user experience. For this reason we provide detailed performance measurements for all 16 of the products tested this month.

THE TEST SET-UP

No major modifications were made to the test set-up, and as usual the full methodology can be found at http://www.virusbtn.com/vbspam/methodology/. In this test developers were offered the option of receiving information on the original sender's IP address and HELO/EHLO domain during the SMTP transaction, thus emulating a real environment where many messages are blocked because of the IP addresses and/or the domains of the senders. However, none of the developers chose to make use of the option on this occasion.

As in previous tests, the products that needed to be installed on a server were installed on a *Dell PowerEdge R200*, with a 3.0GHz dual core processor and 4GB of RAM. The *Linux* products ran on *SuSE Linux Enterprise Server 11*; the *Windows Server* products ran on either the 2003 or the 2008 version, depending on which was recommended by the vendor. (It should be noted that most products run on several different operating systems.)

To compare the products, we calculate a 'final score', defined as the spam catch (SC) rate minus three times the false positive (FP) rate. Products earn VBSpam certification if this value is at least 96%:

$$SC - (3 \times FP) \geq 96\%$$

THE EMAIL CORPUS

The test ran from 6pm GMT on 9 February 2010 until 7am on 1 March 2010, with an unscheduled break between 17 and 22 February when problems beyond our control left us without reliable information to base the test results on. The corpus contained 254,407 emails: 2,458 ham messages and 251,949 spam messages, where the latter consisted of 237,783 messages provided by Project Honey Pot and 14,166 messages sent to legitimate @virusbtn.com addresses.

Some new email discussion lists were added to the ham set and, as in previous tests, emails that claimed to be sent from @virusbtn.com addresses were removed from the test set. (Note that this check was only applied on the MAIL FROM, not on the email headers, and in future tests, these emails will not be removed from the test set.)

For each product, no more than four false positives were counted per sender. The 'image spam' and 'large spam' categories referenced in the test results are, respectively, spam messages containing at least one inline image, and those with a body size of over 50,000 bytes.

BitDefender Security for Mail Servers 3.0.2

SC rate (total): 97.96%
SC rate (Project Honey Pot corpus): 98.68%
SC rate (VB spam corpus): 89.85%
SC rate (image spam): 96.33%
SC rate (large spam): 93.16%
FP rate: 0.04%
Final score: 97.84%

Most products in this month's test saw a slight reduction in their spam catch rate, and this included *BitDefender*. However, *BitDefender* more than made up for this by missing just a single legitimate email out of 2,400. The product easily earns its sixth VBSpam award in a row.

(*Note: On careful investigation of the previous test results – see VB, January 2010, p.23 – it was discovered that BitDefender's false positive score should*

	True negative	False positive	FP rate	False negative	True positive	SC rate	Final score
BitDefender	2457	1	0.04%	5144	246805	97.96%	97.84%
FortiMail	2453	5	0.20%	5401	246548	97.86%	97.26%
Kaspersky	2449	9	0.37%	5148	246801	97.96%	96.85%
M86 MailMarshal	2454	4	0.16%	1717	250232	99.32%	98.84%
McAfee Email Gateway	2438	20	0.81%	2208	249741	99.12%	96.69%
McAfee EWSA	2456	2	0.08%	4281	247668	98.30%	98.06%
MessageStream	2452	6	0.24%	2762	249187	98.90%	98.18%
MS Forefront	2454	4	0.16%	602	251347	99.76%	99.28%
MXTools	2458	0	0.00%	4922	247027	98.05%	98.05%
Sophos	2454	4	0.16%	1787	250162	99.29%	98.81%
SPAMfighter	2446	12	0.49%	5099	246850	97.98%	96.51%
SpamTitan	2452	6	0.24%	1858	250091	99.26%	98.54%
Sunbelt VIPRE	2444	14	0.57%	4004	247945	98.41%	96.70%
Symantec Brightmail	2456	2	0.08%	2263	249686	99.10%	98.86%
Webroot	2456	2	0.08%	3192	248757	98.73%	98.49%
Spamhaus	2458	0	0.00%	5529	246420	97.81%	97.81%

have been 15, rather than the reported 17. This gave the product a FP rate of 0.534% and a final score of 96.51%.)

Fortinet FortiMail

SC rate (total): 97.86%
SC rate (Project Honey Pot corpus): 98.14%
SC rate (VB spam corpus): 93.03%
SC rate (image spam): 95.66%
SC rate (large spam): 93.50%
FP rate: 0.20%
Final score: 97.26%

Fortinet's *FortiMail* appliance has been filtering *VB* email without any problems for five tests in a row. A lower false positive rate on this occasion saw the product's final score improve a little to fully merit its fifth consecutive VBSpam award.

Kaspersky Anti-Spam 3.0

SC rate (total): 97.96%
SC rate (Project Honey Pot corpus): 98.47%
SC rate (VB spam corpus): 89.37%
SC rate (image spam): 97.56%
SC rate (large spam): 94.23%
FP rate: 0.37%
Final score: 96.85%

Kaspersky Anti-Spam did not miss a single legitimate email in the previous test but, thanks to a rather low spam catch rate, the product failed to win a VBSpam award. The product's developers used the feedback from the last test to improve its heuristics-based botnet traffic detection. Indeed, the spam catch rate saw a significant increase and although there were some false positives this time, the product easily reclaimed its VBSpam award.

M86 MailMarshal SMTP

SC rate (total): 99.32%
SC rate (Project Honey Pot corpus): 99.46%
SC rate (VB spam corpus): 96.95%
SC rate (image spam): 99.83%
SC rate (large spam): 98.86%
FP rate: 0.16%
Final score: 98.84%

M86 MailMarshal SMTP – which was tested on *Windows Server 2003*, but which also runs on *Windows Server 2008* – was the highest ranking product in the last test. On this occasion the product saw both its SC rate and its FP rate worsen a little, but not in a significant way, and with the third highest final score, the product is once again ranked highly in this test.

McAfee Email Gateway (formerly IronMail)

SC rate (total): 99.12%
SC rate (Project Honey Pot corpus): 99.37%
SC rate (VB spam corpus): 95.02%
SC rate (image spam): 99.19%
SC rate (large spam): 98.16%
FP rate: 0.81%
Final score: 96.69%

McAfee's *Email Gateway* appliance caught well over 99% of all spam for the fourth time in a row and the product wins its fourth consecutive VBSpam award. However, *Email Gateway* false positived on more legitimate emails than any other product – it had particular difficulties with emails from Eastern European and Asian countries – and there is definitely room for improvement in this area.

McAfee Email and Web Security Appliance

SC rate (total): 98.30%
SC rate (Project Honey Pot corpus): 98.75%
SC rate (VB spam corpus): 90.82%
SC rate (image spam): 92.89%
SC rate (large spam): 94.01%
FP rate: 0.08%
Final score: 98.06%

McAfee's *Email and Web Security Appliance* performed a little disappointingly in the last test, displaying a higher false positive rate than in earlier tests. Further investigation determined that this had been caused by the product sending some temporary failure responses over the Christmas period. The rules of the test stipulate that email that has not reached the back-end MTA one hour after its original delivery will be considered to have been marked as spam, but it is fair to say that in a real situation – with most legitimate senders resending over longer periods of time – this would have led to short delays in email delivery and probably not to false positives. Happily, the product has been working steadily since, missing just two legitimate emails on this occasion, and with an impressive spam catch rate.

MessageStream

SC rate (total): 98.90%
SC rate (Project Honey Pot corpus): 99.19%
SC rate (VB spam corpus): 94.11%
SC rate (image spam): 98.02%
SC rate (large spam): 96.88%
FP rate: 0.24%
Final score: 98.18%

The *MessageStream* hosted solution was one of the first products to join the VBSpam tests and the developers' confidence in their product has proven to be justified time and time again. With another good spam catch rate and missing just a handful of legitimate emails, the product fully deserves a VBSpam award.

Microsoft Forefront Protection 2010 for Exchange Server

SC rate (total): 99.76%
SC rate (Project Honey Pot corpus): 99.86%
SC rate (VB spam corpus): 98.06%
SC rate (image spam): 99.86%
SC rate (large spam): 99.04%
FP rate: 0.16%
Final score: 99.28%

One of the top performers in the previous test, *Microsoft*'s *Forefront Protection 2010 for Exchange Server* saw its performance improve even further and the product outperformed its competitors in all spam categories. Thanks to just four false positives, *Forefront* was the only product to achieve a final score of over 99%.

MXTools Reputation Suite

SC rate (total): 98.05%
SC rate (Project Honey Pot corpus): 98.66%
SC rate (VB spam corpus): 87.70%
SC rate (image spam): 96.79%

	Project Honey Pot spam		VB spam corpus		Image spam*		Large spam*	
	False negative	SC rate	False negative	SC rate	False negative	SC rate	False negative	SC rate
BitDefender	3142	98.68%	1438	89.85%	282	96.33%	186	93.16%
FortiMail	4413	98.14%	988	93.03%	334	95.66%	177	93.50%
Kaspersky	3642	98.47%	1506	89.37%	188	97.56%	157	94.23%
M86 MailMarshal	1285	99.46%	432	96.95%	13	99.83%	31	98.86%
McAfee Email Gateway	1503	99.37%	705	95.02%	62	99.19%	50	98.16%
McAfee EWSA	2980	98.75%	1301	90.82%	547	92.89%	163	94.01%
MessageStream	1927	99.19%	835	94.11%	152	98.02%	85	96.88%
MS Forefront	327	99.86%	275	98.06%	11	99.86%	26	99.04%
MXTools	3179	98.66%	1743	87.70%	247	96.79%	183	93.27%
Sophos	1120	99.53%	667	95.29%	79	98.97%	111	95.92%
SPAMfighter	3956	98.34%	1143	91.93%	151	98.04%	97	96.44%
SpamTitan	1280	99.46%	578	95.92%	40	99.48%	41	98.49%
Sunbelt VIPRE	3368	98.58%	636	95.51%	357	95.36%	132	95.15%
Symantec Brightmail	1397	99.41%	866	93.89%	89	98.84%	101	96.29%
Webroot	2709	98.86%	483	96.59%	41	99.47%	60	97.79%
Spamhaus	3530	98.52%	1999	85.89%	252	96.72%	193	92.91%

*There were 7,691 spam messages containing images and 2,721 considered large; the two are not mutually exclusive.

SC rate (large spam): 93.27%

FP rate: 0.00%

Final score: 98.05%

MXTools Reputation Suite combines *Spamhaus ZEN + RBL* (see below) with the *SURBL* URI blacklist and the *Server Authority* domain reputation service. I stated in the last review that the performance of the latter two depends on the way URIs are detected in emails. We have since found a bug in the script that detects URIs which caused the product to miss several domains, in particular most .cn domains.

Fixing this bug (while also realizing that a lot of spammers have recently moved from .cn to .ru domains) saw the suite's performance improve to a spam catch rate of well over 98%. Equally impressively, there were no false positives this time. The relatively low spam catch rate on the VBSpam corpus suggests that those employing the suite in a real situation would do well to run a filter on the full content of the email too, but nevertheless the suite is the deserving winner of another VBSpam award.

Sophos Email Appliance

SC rate (total): 99.29%

SC rate (Project Honey Pot corpus): 99.53%

SC rate (VB spam corpus): 95.29%

SC rate (image spam): 98.97%

SC rate (large spam): 95.92%

FP rate: 0.16%

Final score: 98.81%

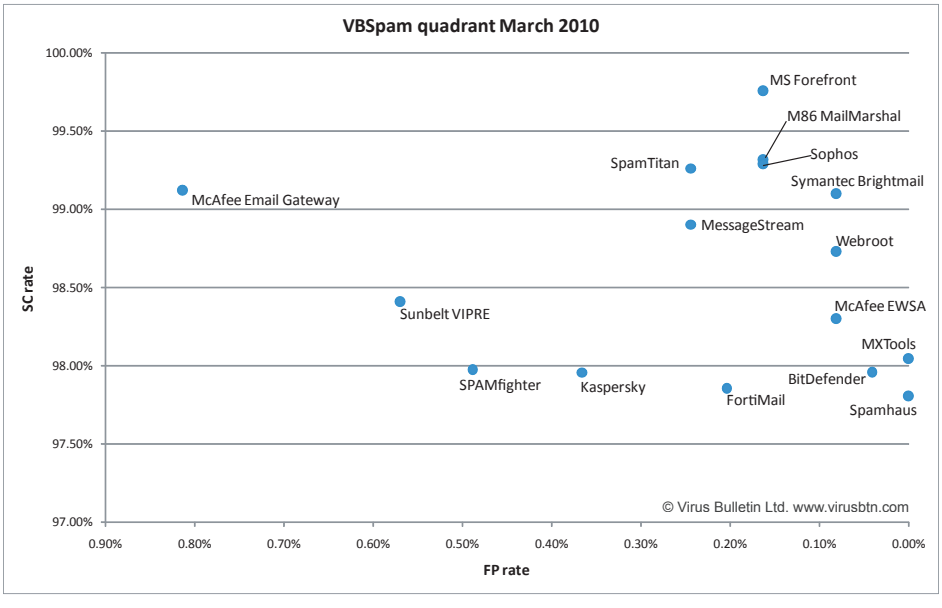
Sophos has been active in the anti-virus industry for a quarter of a century and, like most of its competitors, has been offering anti-spam solutions for quite some time too. *Sophos Email Appliance* is a hardware solution that filters inbound and, optionally, outbound email for spam and malware, as well as offering data protection and email encryption. These and other policies can be configured using a simple web interface, which I found easy to work with; the various reports and trends on email traffic will no doubt help experienced administrators to fine-tune the settings so that they work even better in their organizations.

But even without an administrator's intervention the product worked very well, achieving the third-highest spam catch rate with only a handful of false positives and one of the better final scores. If there was anything that needed to be improved, it would be the product's performance on large emails, but even here it caught the vast majority of spam.

SPAMfighter Mail Gateway

SC rate (total): 97.98%
SC rate (Project Honey Pot corpus): 98.34%
SC rate (VB spam corpus): 91.93%
SC rate (image spam): 98.04%
SC rate (large spam): 96.44%
FP rate: 0.49%
Final score: 96.51%

SPAMfighter Mail Gateway saw a slight improvement in its spam catch rate compared to the previous test, while its false positive rate was about the same; with such a performance the product easily wins another VBSpam award. It was good to see *SPAMfighter*'s performance on both large and image spam improve significantly, and hopefully next time the false positive rate will improve too: while this mostly concerned newsletters (arguably emails that are less likely to be missed by end-users), there is still room for some improvement here.



SpamTitan

SC rate (total): 99.26%
SC rate (Project Honey Pot corpus): 99.46%
SC rate (VB spam corpus): 95.92%
SC rate (image spam): 99.48%
SC rate (large spam): 98.49%
FP rate: 0.24%
Final score: 98.54%

Like many products this month, *SpamTitan* had a lower spam catch rate than in the previous test – when it caught more spam than any other product – but it also saw its false positive rate reduced. This resulted in another impressive final score, putting the product firmly in position as one of this month's top five performers.

Sunbelt VIPRE Email Security

SC rate (total): 98.41%
SC rate (Project Honey Pot corpus): 98.58%
SC rate (VB spam corpus): 95.51%
SC rate (image spam): 95.36%
SC rate (large spam): 95.15%
FP rate: 0.57%
Final score: 96.70%

Like many products in this test, *Sunbelt*'s *VIPRE* combines a slightly lower spam catch rate with a slightly lower false positive rate. The latter in particular still leaves some room for improvement, but it should also be noted that the product had a consistently high spam catch rate, even during periods when most other products saw their performance temporarily drop. This suggests that new spam campaigns are no problem for *VIPRE*.

Symantec Brightmail Gateway 9.0

SC rate (total): 99.10%
SC rate (Project Honey Pot corpus): 99.41%

SC rate (VB spam corpus): 93.89%
SC rate (image spam): 98.84%
SC rate (large spam): 96.29%
FP rate: 0.08%
Final score: 98.86%

Symantec Brightmail Gateway debuted in the previous test with an impressive performance and the third best final score. On this occasion we tested a new version of the product (a virtual appliance) which performed even better: like most products, its spam catch rate was slightly lower on this occasion, but this was more than made up for by the fact that it missed just two legitimate emails, resulting in the second best final score overall.

Webroot E-Mail Security SaaS

SC rate (total): 98.73%
SC rate (Project Honey Pot corpus): 98.86%
SC rate (VB spam corpus): 96.59%
SC rate (image spam): 99.47%
SC rate (large spam): 97.79%
FP rate: 0.08%
Final score: 98.49%

Webroot's hosted anti-spam solution has had a consistently high spam catch rate ever since joining the very first VBSpam test. In the past, the product has suffered from more false positives than average, but the developers must have worked hard on this and the product missed only two legitimate emails this time. If this is the reason fewer spam messages were caught, then I would say it's been worth it, as the product saw its final score improve.

Spamhaus ZEN plus DBL

SC rate (total): 97.81%
SC rate (Project Honey Pot corpus): 98.52%
SC rate (VB spam corpus): 85.89%
SC rate (image spam): 96.72%
SC rate (large spam): 92.91%
FP rate: 0.00%
Final score: 97.81%

As in the previous test, the IP address of every incoming email was checked against the *Spamhaus ZEN* DNS blacklist, while domain checks were performed against the new *Spamhaus DBL* blacklist. Once again, this resulted in a very good spam catch rate and again there were no false positives. While it is probably not a good idea to use a DNS blacklist as a standalone spam filter, with *Spamhaus* one can at least be sure that the vast majority of spam is blocked at an early stage.

Products ranked by final score	Final score
MS Forefront	99.28%
Symantec Brightmail	98.86%
M86 MailMarshal	98.84%
Sophos	98.81%
SpamTitan	98.54%
Webroot	98.49%
MessageStream	98.18%
McAfee EWSA	98.06%
MXTools	98.05%
BitDefender	97.84%
Spamhaus	97.81%
FortiMail	97.26%
Kaspersky	96.85%
Sunbelt VIPRE	96.70%
McAfee Email Gateway	96.69%
SPAMfighter	96.51%

CONCLUSION

For a few tests in a row we have been adding to the ham corpus the traffic of several email discussion lists. In the next test, we plan to take this one step further: we will use the emails sent to the lists, but rewrite the headers as well as the IP address and HELO/EHLO domain in such a way that, to the products in the test, it will look as if the emails have been sent directly to us rather than via the list server. This is not a trivial thing to do and certainly doesn't work for all mailing lists, but tests run over the past weeks show that it works well and that it creates a varied ham corpus.

We also plan to remove the *VB* corpora from the test. Over the past year our own email has given us a very realistic email stream to test against, but the fact that we have been unable to share full details of incorrectly classified emails with developers has become increasingly frustrating for all involved. Although developers have rarely questioned our decisions, in order for them to be able to improve their products – one of the most important aspects of the anti-spam tests – they need to have access to the full emails.

The products' performance on the *VB* spam and ham corpora will be included in the next report. However, these results will not count towards their final score.

The next test is set to run throughout April with the deadline for product submission being 26 March 2010; any developers interested in submitting a product should email martijn.grooten@virusbtn.com.

END NOTES & NEWS

The 7th Annual Enterprise Security Conference will take place 3–4 March 2010 in Kuala Lumpur, Malaysia. For details see http://www.acnergy.com/EntSec2010.htm.

Security Summit Milan takes place 16–18 March 2010 in Milan, Italy (in Italian). For details see https://www.securitysummit.it/.

The 11th annual CanSecWest conference will be held 22–26 March 2010 in Vancouver, Canada. For more details see http://cansecwest.com/.

The MIT Spam Conference 2010 is scheduled to take place 25–26 March 2010. For details see http://projects.csail.mit.edu/spamconf/.

Black Hat Europe 2010 takes place 12–15 April 2010 in Barcelona, Spain. For details see http://www.blackhat.com/.

The New York Computer Forensics Show will be held 19–20 April 2010 in New York, NY, USA. For more information see http://www.computerforensicshow.com/.

Infosecurity Europe 2010 will take place 27–29 April 2010 in London, UK. For more details see http://www.infosec.co.uk/.

The 19th EICAR conference will be held 10–11 May 2010 in Paris, France with the theme 'ICT security: quo vadis?'. For more information see http://www.eicar.org/conference/.

The fourth annual Counter-eCrime Operations Summit (CeCOS IV) will take place 11–13 May 2010 in São Paulo, Brazil. For details see http://www.apwg.org/events/2010_opSummit.html.

NISC11 will be held 19–21 May 2010 in St Andrews, Scotland. Interest in attending can be registered at http://nisc.org.uk/.

The International Secure Systems Development Conference (ISSD) takes place 20–21 May 2010 in London, UK. For details see http://issdconference.com/.

CARO 2010, the 4th International CARO workshop will take place 26–27 May 2010 in Helsinki, Finland. The workshop will focus on the topic of 'Big Numbers'. For more information see http://www.caro2010.org/.

CSI SX – Security for Business Agility takes place 26–27 May 2010 in San Francisco, CA, USA. The event will address the challenges of managing security in an increasingly mobile business environment. For details see http://www.csisx.com/.

Security Summit Rome takes place 9–10 June 2010 in Rome, Italy (in Italian). For details see https://www.securitysummit.it/.

The 22nd Annual FIRST Conference on Computer Security Incident Handling takes place 13–18 June 2010 in Miami, FL, USA. For more details see http://conference.first.org/.

The Seventh International Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA) will take place 8–9 July 2010 in Bonn, Germany. For more information see http://www.dimva.org/dimva2010/.

CEAS 2010 – the 7th annual Collaboration, Electronic messaging, Anti-Abuse and Spam Conference – will be held 13–14 July 2010 in Redmond, WA, USA. A call for papers has been issued, with a deadline for submissions of 26 March. For details see http://ceas.cc/.

Black Hat USA 2010 takes place 24–29 July 2010 in Las Vegas, NV, USA. DEFCON 18 follows the Black Hat event, taking place 29 July to 1 August, also in Las Vegas. For more information see http://www.blackhat.com/ and http://www.defcon.org/.

The 19th USENIX Security Symposium will take place 11–13 August 2010 in Washington, DC, USA. For more details see http://usenix.org/.

VB2010 will take place 29 September to 1 October 2010 in Vancouver, Canada. The deadline for the call for papers for VB2010 is 5 March – see http://www.virusbtn.com/conference/vb2010/. For details of sponsorship opportunities and any other queries relating to VB2010, please contact 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/