NOVEMBER 2008

virus BULLETIN

Fighting malware and spam

CONTENTS

2 **COMMENT**
Co-operation is the only way

3 **NEWS**
Job cuts in the offing at Symantec
Plug pulled on dodgy registrar?

3 **VIRUS PREVALENCE TABLE**

MALWARE ANALYSES
4 XXX racted
6 Your filters are bypassed: Rustock.C in the kernel

12 **OPINION**
Family matters

13 **CONFERENCE REPORT**
The Ottawa rules

17 **PRODUCT REVIEW**
DriveSentry Desktop 3.1/3.2 & GoAnywhere 1.0.2/2.0

21 **END NOTES & NEWS**

ISSN 1749-7027

IN THIS ISSUE

A JOKER IN THE PACK

Last month saw the annual three-day work-rest-and-play marathon (without so much of the rest) that is the *VB* conference. A slowly recovering Helen Martin reports on VB2008.
page 13

ON GUARD

John Hawes looks at two products from *DriveSentry* – part of a growing 'new breed' of security products which focus less on the traditional arts of the anti-malware world, and aim instead to protect systems by preventing unauthorized software from performing any potentially dangerous activity.
page 17

vbSpam supplement

This month: anti-spam news and events, and Terry Zink completes his series of articles on backscatter with a look at Bounce Address Tag Validation.

'An essential force in the fight against online crime is that of law enforcement.'

Martijn Grooten, Virus Bulletin

CO-OPERATION IS THE ONLY WAY

The 'Security in banking' discussion forum held at the close of VB2008 last month had been planned for many months – the original idea taking shape at a time when banks seemed healthy businesses, taking care to look after their customers' money. But come the first days of October many leading banks saw their stock prices plummet; some even faced bankruptcy.

To an outsider, the topic of online banking crime might have seemed trivial when compared to the billions the banks were losing every day. Of course, it isn't. As many experts have pointed out, losses and gains on the stock markets have a lot to do with trust: do traders trust a bank to do well in the near future? A bank whose accounts are compromised by crooks in a faraway country may not seem very trustworthy. Moreover, the banking crisis has led to an increase in the number of online scams targeting banks. A report by *MessageLabs* indicates that the number of phishing scams has more than doubled in the past month (http://www.messagelabs.co.uk/resources/press/19846), and the FTC has seen fit to issue a warning to consumers (http://www.ftc.gov/bcp/edu/pubs/consumer/alerts/alt089.shtm).

This doesn't mean that banks haven't been working hard to secure their systems, or that security vendors haven't put in the effort to protect their customers. It doesn't even mean that most users are still unaware of the dangers of online fraud: many phishing scams these days even contain warnings about the dangers of phishing in order to make them look more legitimate.

But the fight against online crime won't be won solely by security products and user education; an essential third force in the fight against online crime is that of law enforcement. Unfortunately, prosecutions of cybercriminals are relatively rare and usually involve kids who broke into their school's computers.

It came as a welcome change, therefore, to hear of the recent arrests, thanks to cross-border co-operation between police forces, of three individuals involved in phishing scams in Russia and Ukraine.

The investigation began in the Netherlands in March 2007, when online banking customers of *ABN AMRO* were targeted in a phishing scam. After complaints from many customers the bank called in the Dutch police, whose Team High Tech Crime (THTC) took on the investigation, leading to the arrest of 14 money mules late in 2007. Further probing led investigators to believe that the ringleaders of the scam were based in Russia and Ukraine, and the case files were translated and handed over to local police – the eventual outcome of which was the arrest of the three suspects last month.

Not only should the efforts of the various law enforcement agencies be applauded – the crossing of borders is notoriously tricky where law enforcement is concerned – but *ABN AMRO* should be commended for its openness; many banks choose to remain silent about such attacks, for fear of damage to their reputations.

ABN AMRO was just one of many Western European banks suspected to have been targeted by the same phishing gang, all of which will benefit from the arrests. This clearly shows the importance of co-operation in the fight against online fraud – as was also demonstrated recently by researchers at the University of Cambridge who estimated that the lack of data-sharing between 'take-down companies' – the companies hired by banks to take down phishing sites – costs the banking industry at least $350 million a year (see http://www.lightbluetouchpaper.org/2008/10/16/non-cooperation-in-the-fight-against-phishing/).

Co-operation does not end here though: end-users can contribute to the fight against cybercrime by reporting any online crime they have spotted. To help both home and business users in reporting cybercrime, *VB* has put together a collection of relevant links and resources and made them available at: http://www.virusbtn.com/resources/cybercrime/index.

It is unlikely that online crime will ever disappear; indeed, in the foreseeable future it is likely to increase. At the same time, online banking is a convenient and generally secure way of managing bank accounts. But to prevent the large amounts of taxpayers' money that have been pumped into banks recently from ending up in the hands of criminals, co-operation is the only way to go.

NEWS

JOB CUTS IN THE OFFING AT SYMANTEC

Symantec chiefs announced this week that the company is in the process of a 'reduction in force' alongside other budget-scrimping measures that include cutting down on travel expenses and other discretionary purchases.

Speaking at the company's second quarter 2009 earnings conference call, *Symantec*'s CFO James Beer said that the company plans to cut its head count budget by 4.5%. Although he wouldn't be drawn on the number of job losses anticipated, he revealed that the company's current work force stands at just under 18,000 employees. *Symantec* has also recently outsourced parts of its internal IT and finance back office operations and, Beer revealed, is in the process of outsourcing its European manufacturing operations from Ireland to the Czech Republic.

While, according to CEO John Thompson, the company generated year-over-year growth in revenue, demonstrated solid progress on its goals for operating margins and delivered strong earnings growth, the second quarter saw costs rise as the company completed its acquisition of spyware specialist *PC Tools* in October.

Earlier in October the firm, which in the past has had something of a reputation for making regular acquisitions, announced a definitive agreement to acquire software-as-a-service security specialist *MessageLabs* – paying a purchase price of approximately $695 million in cash. The deal is not expected to close until the back end of the December quarter.

PLUG PULLED ON DODGY REGISTRAR?

ICANN, the organization responsible for managing the assignment of domain names and IP addresses, has announced its intention to pull the plug on Estonian domain registrar EstDomains – long known to be favoured by cybercriminals for their domain registrations.

In an official letter dated 28 October, ICANN advised EstDomains that the official ICANN Registration Accreditation Agreement with the company would be terminated on 12 November 2008, citing company president Vladimir Tsastsin's conviction of credit card fraud, money laundering and document forgery as the reason for the termination. However, a response from EstDomains has since been received by ICANN, advising the organization that Tsastsin resigned from his post prior to his conviction and has since been replaced – thus appealing to ICANN to reconsider the termination of the agreement. ICANN has put the termination process on hold while it investigates.

Experts estimate that tens of thousands of malicious domains have been registered through EstDomains including sites used in drive-by-downloads, botnet command-and-control servers, spammed domains and so on.

Prevalence Table – September 2008

Malware	Type	%
Agent	Trojan	32.49%
Inject	Trojan	20.33%
Autorun	Worm	11.51%
Hijack	Trojan	8.49%
Suspect packers	Misc	5.94%
Goldun	Trojan	5.07%
NetSky	Worm	2.18%
Zbot	Trojan	1.81%
Downloader-misc	Trojan	1.56%
Buzus	Trojan	1.50%
Virut	Virus	1.27%
PWS-misc	Trojan	0.75%
Mytob	Worm	0.73%
Dropper-misc	Trojan	0.72%
Bifrose/Pakes	Trojan	0.64%
Small	Trojan	0.59%
Bagle	Worm	0.59%
Crypt	Trojan	0.43%
Mydoom	Worm	0.35%
Basine	Trojan	0.34%
Cutwail/Pandex/Pushdo	Trojan	0.32%
Zafi	Worm	0.26%
Delf	Trojan	0.21%
Grew	Worm	0.20%
Iframe	Exploit	0.18%
OnlineGames	Trojan	0.17%
Lineage/Magania	Trojan	0.17%
Parite	Worm	0.16%
Heuristic/generic	Trojan	0.15%
Zlob/Tibs	Trojan	0.13%
Mdropper	Trojan	0.10%
Qhost	Trojan	0.10%
Alman	Worm	0.09%
Others[1]		0.48%
Total		100.00%

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

MALWARE ANALYSIS 1

XXX RACTED

Peter Ferrie
Microsoft, USA

We have reached the last in the collection of viruses created by the writer 'fakedmnded' in the EOF-rRlf-DoomRiderz virus zine (see also *VB* September 2008, p.4, and October 2008, p.4), although it is not the last entry in the series from the virus zine itself. This one is called W32/Exract.

STUPID IS AS STUPID DOES

The virus begins by getting the address of the IsDebuggerPresent() API and then calling it. If a debugger is present the virus exits. The virus also checks for alterations within the code that has just run. This is probably intended to detect breakpoints, but in normal circumstances, there wouldn't be any breakpoints left at that point. The routine also contains some dead code, which would have called the IsDebuggerPresent() API, and checked once again for alterations. It is fortunate, in a way, that the code doesn't run because, given the way in which the code is structured, the second check would always fail, and the virus code would crash shortly afterwards.

It is possible that the dead code is intended as a decoy, to tempt someone into placing a breakpoint there, which would lead to the virus being able to run freely. However, experience suggests that it is best not to assume something smart where something stupid is more likely.

After some further checks for alterations, including one in a location that has already been checked, we see some familiar code.

I'M A LOCAL

The virus stores the selector of the local descriptor table onto the stack, and then reads four bytes and checks if the result is non-zero. The result should always be non-zero because the location on the stack holds the previous stack frame when the process started, which is always an address above the 64 KB boundary. As a result, the top half of the stack frame will remain untouched and non-zero.

This might be an anti-emulator trick for an emulator that stores four bytes instead of two. However, it seems more likely that what the virus author had in mind was to read only two bytes and detect whether the local descriptor table (LDT) is in use, but had to reverse the condition because of the extra bytes that the virus reads. The use of the LDT is a characteristic of virtual machines such as *VMware* and *VirtualPC*, along with *Norman*'s *SandBox*.

IT'S PAYBACK

The virus carries two payloads. The first triggers on the 12th of any month. At that time, the virus displays a message box whose title is 'Sorry Unable to extract the file!' with the message body:

```
Error 617573 :Shareware period has been elapsed!

For more info search for 'Fakedminded' on google ,and
play warcraft too!
```

The author spelled his name correctly this time. I did as suggested, and searched for 'fakedminded' on *Google*. Funnily enough, none of the returned pages belonged to him.

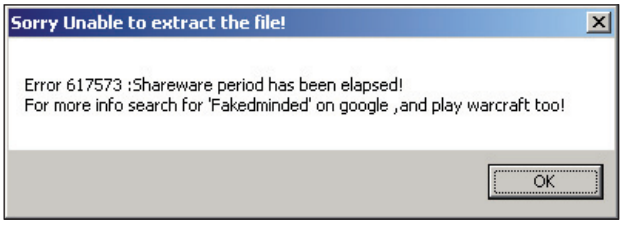
OPEN SESAME

The second payload triggers on the 4th of October. At that time, the virus attempts to drop a file called 'kloka.vbs' into the root directory of the C: drive. This action is disallowed by default under *Windows Vista*.

The virus attempts to create the registry key 'HKEY_CLASSES_ROOT\.sy64', and to set its default value to 'DOS1234'. The virus also attempts to create the registry key 'HKEY_CLASSES_ROOT\DOS1234\shell\open\command' and to set the default value to point to the 'kloka.vbs' file. However, the creation of registry keys in that location is disallowed by default under *Windows Vista*. There is also a bug in the registry code, which appears twice: if the registry value cannot be set, then the virus does not close the registry handle. The result is a handle leak if an error occurs.

The idea of that registry modification is to register a new suffix. Thereafter, executing a file whose suffix is '.sy64' will cause the script file to run. The virus attempts to produce this effect automatically, by creating a file called 'sysvb.sy64' directly in the Start Menu at 'C:\Documents And Settings\All Users\Start Menu\Programs\Startup' and 'C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup'. This action is also disallowed by default under *Windows Vista*.

If the 'kloka.vbs' file were to be executed, it would attempt to access the *Kaspersky Lab* website (www.kaspersky.com) once every ten seconds, in an infinite loop.



BUT WAIT, THERE'S MORE

Now we reach the main part of the virus. The virus opens its own file, requests the file size (which is not constant and might be very large, see below), and then allocates some memory to hold a copy of the entire file. There is a bug in this code, which is that the memory is never freed. Amazingly, the virus is really only interested in the file offset of the original end of the file.

The virus checks if its filename ends with 'bye'. There is a bug here, which is that the comparison is case-sensitive. If the filename does end with 'bye', then the virus attempts to delete the file 'C:\Program Files\Common Files\hushabye.exe'. However, the deletion of the file in that location is disallowed by default under *Windows Vista*.

The virus attempts to create the registry key 'HKEY_CLASSES_ROOT\.err64', and to set its default value to 'Coconest'. The virus also attempts to create the registry key 'HKEY_CLASSES_ROOT\Coconest\shell\open\command' and to set the default value to point to the 'C:\Program Files\Common Files\hushabye.exe' file. As before, the creation of registry keys in that location is disallowed by default under *Windows Vista*. The bugs that result in a possible handle leak (if an error occurs) are also present here.

Thereafter, executing a file whose suffix is '.err64' will cause the exe file to run. The virus attempts to produce this effect automatically, by creating a file called 'sysCheckup.err64' in the Start Menu, in the same location as 'sysvb.sy64'. Once again, this action is disallowed by default under *Windows Vista*. The virus then attempts to copy itself as 'hushabye.exe' to the 'Common Files' directory. This action is also disallowed by default under *Windows Vista*.

CLIP GO THE SHEARS

Once the installation is complete, and if the filename ends with 'bye', then the virus opens the clipboard and saves the 'handle' that is returned. In fact, what is returned is not a handle, but a flag that indicates success or failure. The virus queries the clipboard for a list of files that are currently being copied. If such a list exists, then the virus calculates the length of the clipboard data using a very poorly coded routine. Instead of performing a wcschr() to parse the Unicode characters correctly, the virus performs a byte-level step while comparing words in memory. This can lead to early termination if particular characters are found in the string, such as the Tibetan syllable 'Om'. Meditate on that, grasshopper.

The virus finds the last string in the list, calculates the length in bytes of that string, and then allocates that length. There is a bug in this routine, which is that the buffer is never freed. This can quickly become a problem. Since the code executes in a loop, and if the clipboard is not used for a while, then the same string will be seen repeatedly. This will cause further memory allocations, and eventually exhaust the system resources.

STRING THEORY

The virus copies the string to the newly allocated buffer, and converts it from Unicode to ASCII at the same time. A minor bug exists here, which is that the virus uses the byte count as a character count while copying the string. This results in the virus writing twice as much data as necessary, but is not a problem because the buffer is large enough to hold all of the data.

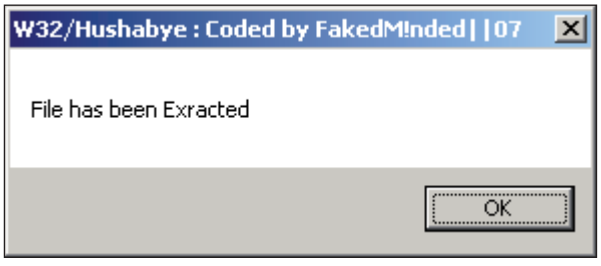
The virus also allocates a buffer to hold a copy of the string. There is a bug in this routine, which is that the buffer is never freed. The virus copies the string to this buffer, and then examines the copied string. If the string contains only a directory name, then the virus will skip the infection. Otherwise, the virus switches to the directory that contains the file, and then examines the filename. If the filename ends with '-packed.exe', then the virus will also skip the infection. This is the 'infection' marker.

INFECTIOUS GROOVES

The virus opens the file to infect, requests the file size, and then allocates some memory to hold a copy of the entire file. There is a bug in this code, which is that the memory is never freed. The virus reads the whole file, and 'encrypts' it (using just a simple XOR with the letter 'X'). Then the virus opens its own file, requests the file size, and then allocates some memory to hold a copy of the entire file. This is despite the fact that another copy of the virus already exists in memory. There is also a bug in this code, which is that the memory is never freed.

The virus creates the new file '<file>-packed.exe', where '<file>' is the name of the file to infect, and then writes the virus body and the encrypted file to it.

The virus converts the pathname to Unicode, and then allocates memory to hold the Unicode string. There is a bug in this routine, which is that the buffer is not freed. The virus constructs a new 'list' of files, which contains only one entry, and then empties the clipboard of all data before assigning the list. This code could be considered to contain two bugs. The first is that all of the data in the clipboard is discarded, instead of only the data of the file list



type (which can simply be replaced without emptying the clipboard at all). The second bug is that the original file list is discarded, leaving only one file to be copied.

CLOSED FOR THE DAY

The virus calls the CloseHandle() API for the 'handle' that was returned by the OpenClipboard() API. Fortunately for the virus author, the handle is treated as invalid by *Windows* and the request is ignored, rather than causing an error. The virus then closes the clipboard using the correct API.

At this point, either no file list exists, or the infection completed successfully. The virus calls the CloseHandle() API (again), for the 'handle' that was returned by the OpenClipboard() API, and also closes the clipboard (again) using the correct API. The virus sleeps for one second, and then resumes from the top of the function where the clipboard is opened again. Such a short delay is a serious bug. The clipboard is a unique resource, so no other applications can use it while the virus has it open. This produces a race condition for users who are trying to copy items. Sometimes it will work, and most times it won't.

-OOPS

If the filename does not end with 'bye', then the virus searches within the pathname for the '-' character. This is supposed to find the '-packed.exe' files, but it has the buggy behaviour of also finding directories that contain the '-' character. This bug affects the first generation code, such as when it is run from the 'EOF-DR-RRLF' directory. If the wrong file is executed, then the virus will decrypt data beyond the end of the buffer and crash.

However, if the '<file>-packed.exe' file is executed, then the virus will decrypt the appended data, create a new '<file>', and then display a message box stating 'File has been Exracted' [*sic*].

The virus does not run the original file. At this point it runs the installation code, as above, that begins by attempting to delete the 'hushabye.exe'. Finally, once installation is complete, the virus exits.

MALWARE ANALYSIS 2

YOUR FILTERS ARE BYPASSED: RUSTOCK.C IN THE KERNEL

Chandra Prakash
Sunbelt Software, USA

Following earlier articles on Rustock.A (see *VB*, September 2006, p.6) and Rustock.C (see *VB* August 2008, p.4), this article describes the step-by-step operational characteristics of Rustock.C in kernel mode from its startup to the point at which its spambot code (botdll) is activated in user mode[1].

Understanding the operational characteristics of Rustock.C through static analysis is a very cumbersome process as it executes after several stages of unpacking. Furthermore, multiple threads are created right from the malware's startup, which increase the complexity of dynamic analysis. The analysis presented here is based on a June 2008 sample.

STAGE 1: UNPACKING

In its initial stage Rustock.C uses a simple XOR algorithm to unpack its code to a designated area. Once unpacking is complete, it transfers control to the unencrypted code as shown below:

```
lea     esp, [esp-4]
mov     dword ptr [esp], offset byte_13000
retn
```

A different sample of Rustock.C demonstrates an anti-debugging trick when Stage 1 unpacking is complete:

```
popad
sub   esp,4     ; Increase current top of stack.
mov   dword ptr [esp],offset rustockC+0x3000
add   esp,4     ; Decrease current top of stack.
push dword ptr [esp-4]
                ; Access to a value beyond current top
                ; of stack. In a debugging session,
                ; this stack location may very well
                ; contain previous register eflags
                ; value stored by debug trace
                ; interrupt. As a result EIP after
                ; 'ret' can point to an invalid
                ; location.
ret
```

This shows that sub-variants of Rustock.C exist with slight differences in operational behaviour.

[1] Unless otherwise stated: Any information on operating system routines or data structures applies to 32-bit *Windows XP SP2*; any reference to ntoskrnl also implies a reference to ntkrnlpa, ntkrnlmp, ntkrpamp; a file-mapped PE image refers to a PE file as on disk; a virtual-mapped PE image refers to a PE image in virtual memory as loaded by the *Windows* loader.

After initial unpacking, one of the first things the malware does is to locate the load address of ntoskrnl via some pointer arithmetic on the interrupt descriptor table (IDT) using the following set of instructions:

```
mov     eax,dword ptr fs:[00000038h] ; Get IDT address.
mov     eax,dword ptr [eax+4]
xor     al,al
find_ntos_base:
sub     eax,100h
cmp     word ptr [eax],5A4Dh
jne     find_ntos_base
```

It then scans an address obtained from the first IDT entry to look for the base address of ntoskrnl. The base address of ntoskrnl is used to scan its export table for the following functions:

```
ExAllocatePool
ExFreePool
ZwQuerySystemInformation
_stricmp
```

These functions are used for unpacking and loading as described in the next sections.

STAGE 2: DECOMPRESSION AND DECRYPTION

In Stage 2, Rustock.C allocates a temporary buffer using ExAllocatePoolWithTag to decompress and decrypt data from stage 1. All memory allocation calls in the kernel are made through this API with the tag name 'Ddk ' (note the space). Decompression is carried out using the apLib algorithm followed by decryption using RC4. These decompression and decryption mechanisms are well documented elsewhere [1, 2].

LOADING

The unpacked data from Stage 2 is the final file-mapped PE image of the driver ready to be loaded. The image is loaded with its image base as the start address of the location from which it was originally unpacked, wiping out Stage 1 decrypted data. Loading is carried out in three steps:

- The PE headers and sections are copied over. The starting virtual address of every section is aligned as per the section alignment.
- The IAT table is patched.
- The relocations are fixed.

Imports are mainly from ntoskrnl and hal.dll, which are obtained via a lookup in the export table using their load information. The load information is obtained through ZwQuerySystemInformation using the SystemModuleInformation class. After the IAT fix up, the relocations are completed in place. Once relocations are completed the MZ and PE signatures are zeroed out to obfuscate the loaded image to prevent its detection by kernel debuggers. It then zeroes out and frees up the temporary buffer which contains the file-mapped PE image.

After the image is virtually mapped, control is transferred to the entry point of the final loaded image as shown below:

```
mov     dword ptr [esp+1Ch],esi     ; ESI has entry point.
popad
jmp     eax
```

The activities of the two threads created at startup and a third thread that is created conditionally (see Figure 1) are described in the following sections.

ACTIVITIES OF THREAD1

Setting up hooks

Thread1 starts by creating a named event handle via ZwCreateEvent with the name \BaseNamedObjects\{C8453B23-1087-27d9-1394-CDBF03EC72D8}. The use of the \BaseNamedObjects directory indicates that this event object is intended to be shared with user mode. It starts by searching the NULL terminated ASCII string 'FATAL_UNHANDLED_HARD_ERROR' in the resource section of ntoskrnl. If the string is found, a page-locking test is performed on the page that contains the string using the pseudo code shown below:

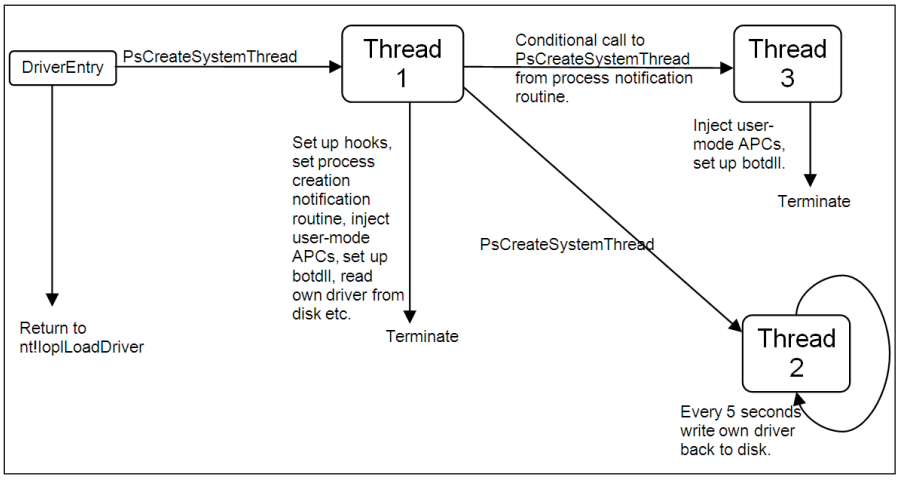


Figure 1.

```
__try
{
    PMDL mdl;
    mdl = IoAllocateMDL(
        vaFatalHandledHardErrorStr,
        0x1b, // NULL terminated length of str.
        0,
        0,
        0);
    MmProbeAndLockPages(
        mdl,
        KernelMode,
        IoAccessRead);
}
__except(EXCEPTION_EXECUTE_HANDLER)
{
    IoFreeMdl(Mdl);
}
```

The call to MmProbeAndLockPages will throw an access violation exception if appropriate access is not granted to the requested pages. In Rustock.C, there is no reference to MDL allocated from IoAllocateMDL, which raises questions as to the purpose of this code here. However, there is a connection to the Rustock.A kiFastCallEntry hook (see below), indicating that Rustock.C is very likely an enhanced version of the Rustock.A code base, and this code is simply left over from its previous version [3, 4].

```
; IA32_SYSENTER_EIP = 0x176
; msr[176] = 806afd59
; 806afd59   e9 ec 2e e6 77 4e 44 4c    ....wNDL
; 806afd61   45 44 5f 48 41 52 44 5f    ED_HARD_
; 806afd69   45 52 52 4f 52 0d 0a 00    ERROR...

806afd59 e9ec2ee677 jmp rustockA+0x4c4a
```

The thread then calls ZwCreateFile to open a handle to the ntdll file using the \SystemRoot\System32\ntdll.dll path. Following this, a call is made to ZwQueryInformationFile to obtain the on-disk size of ntdll using FILE_INFORMATION_CLASS FileStandardInformation. Using the ntdll file size a new buffer is allocated with ExAllocatePoolWithTag and the ntdll file is read off disk using ZwReadFile.

The disk buffer containing ntdll data is then virtually mapped into a new buffer. The new buffer is also allocated via ExAllocatePoolTag, and once it has been virtually mapped, the previous buffer containing the on-disk data is freed. The virtually mapped ntdll is used to obtain the SSDT service number index of hooked Zw functions by searching the function entry in the ntdll export table. When the virtually mapped ntdll is ready, it stores it own load address, size and full driver path in designated memory locations for subsequent use. The self-load information is used to map its own driver into user space as described later. It then sets up its process creation notification routine via PsSetCreateProcessNotifyRoutine and creates a second thread, Thread2, as shown in Figure 1. All sub-keys and values under \registry\machine\system\CurrentControlSet\Enum\Root\LEGACY_<rustockC_driver_name> are deleted recursively.

Rustock.C hooks the registry in a way that has not been seen in previous Rustock variants [5]. It hooks the registry key parse procedure in the kernel that is registered by the configuration manager with the object manager (see below). The parse procedure is employed to parse a registry path in registry-related APIs.

```
_OBJECT_TYPE_INITIALIZER
+0x000 Length             : 0x4c
.
.
+0x030 OpenProcedure      : (null)
+0x034 CloseProcedure     : 0x8056bf9e
nt!CmpCloseKeyObject+0
+0x038 DeleteProcedure    : 0x8056c072
nt!CmpDeleteKeyObject+0
+0x03c ParseProcedure     : 0xf9b4fdd3 <-- Rustock.C
address (normally nt!CmpParseKey).
+0x040 SecurityProcedure  : 0x8056bfd6
nt!CmpSecurityMethod+0
+0x044 QueryNameProcedure : 0x805a935e
nt!CmpQueryKeyName+0
+0x048 OkayToCloseProcedure   : (null)
```

Some more functions, ZwOpenKey and ZwCreateKey, are also hooked. After setting up registry hooks, the malware gets a handle to the directory containing its driver file using ZwCreateFile and that handle is used in ObReferenceObjectByHandle to get a FILE_OBJECT pointer. It then calls IoGetRelatedDeviceObject on the DeviceObject field of the file object to obtain the highest-level device object in the file system filter driver stack. Typically, on machines that support the filter manager, the highest-level device object happens to be the device object of the filter manager driver (FltMgr.sys). Using the highest-level device object the malware walks down the device stack until it finds the lowest-level device object created by the NTFS driver. The device object of the NTFS driver is used to hook its IRP_MJ_CREATE dispatch routine. In one Rustock.C variant, the mechanics of this create hook allowed a copy of its driver from the *Windows* command prompt, but the copy was not the same as the original driver file.

ZwTerminateProcess is then hooked and a function dispatch table is set up, which is used to serve commands from botdll in an unusual way:

```
NTAPI NtTerminateProcess(
       IN HANDLE hProcess,
       IN NTSTATUS ExitCode
);
```

```
00012339 cmp    dword ptr [ebp+0Ch], 0FCC7975Bh
                ; ExitCode parameter contains special
                ; encoded value for botdll and
                ; driver communication.
00012340 jnz    short OrigNtTerminateProcess
       .
       .
OrigNtTerminateProcess:
                ; Normal process termination requests
                ; come here.
000123AF push   dword ptr [ebp+0Ch]
000123B2 push   dword ptr [ebp+8]
000123B5 mov    eax, OrigNtTerminateProcess
000123BA call   dword ptr [eax]
```

The ExitCode parameter of ZwTerminateProcess is set to a specific value that indicates a message from botdll to the driver. The message parameters are encoded in the first hProcess parameter. Normal process termination requests are routed to the original NtTerminateProcess routine address stored in memory as shown above.

Setting up botdll: step 1

Services.exe is used as a goat process for hosting botdll. The process id of the services.exe process is obtained using the SystemProcessAndThreadsInformation class in the ZwQuerySystemInformation call. This process id is used to get the EPROCESS object associated with services.exe. The EPROCESS object is used in the KeAttachProcess call to attach to the virtual address space of services.exe. Then Rustock.C maps its own driver's PE image into the services.exe address space using the IoAllocateMdl, MmBuildMdlForNonPagedPool, MmMapLockedPages sequence of calls. By mapping its own driver image in user space, the malware makes its code and data available to user-mode processes, as described later in this section.

Before calling KeDetachProcess, Rustock.C calls NtSetInformationProcess on services.exe with the PROCESS_INFORMATION_CLASS parameter as ProcessExecuteFlags(0x22) with mask value MEM_EXECUTE_OPTION_ENABLE(0x2). The purpose of this call is to disable the no-execute (NX) bit for DEP data pages [6]. The malware then gets information of all services.exe threads using the SystemProcessAndThreadsInformation class in ZwQuerySystemInformation called earlier and sends an asynchronous procedure call (APC1) to each of the threads. The APC mechanism is designed to execute a function in the context of a target thread. The API calls used for APC are KeInitializeApc and KeInsertQueueApc:

```
NTKERNELAPI
VOID
KeInitializeApc (
    IN PRKAPC Apc,
    IN PKTHREAD Thread,
    IN KAPC_ENVIRONMENT Environment,
    IN PKKERNEL_ROUTINE KernelRoutine,
    IN PKRUNDOWN_ROUTINE RundownRoutine OPTIONAL,
    IN PKNORMAL_ROUTINE NormalRoutine OPTIONAL,
    IN KPROCESSOR_MODE ApcMode,
    IN PVOID NormalContext
    )
```

The NormalRoutine and NormalContext parameters are the user-mode virtual addresses of the APC1 start routine and its context respectively. Note the values for these user-mode virtual addresses are set earlier by mapping the malware's own kernel PE image into user space. The KernelRoutine parameter in KeInitializeApc is the address of a function in kernel space that frees up the APC object (first parameter) allocated from a non-paged pool. The primary purpose of the APC1 call is to set up the import address table of function names referenced in the NormalContext field:

```
LoadLibraryA
GetProcAddress
SetEvent
Init
CreateThread
SleepEx
```

The virtual addresses of these functions are resolved using the load address of kernel32.dll from dll load information stored in the process environment block (PEB). The address of the PEB is obtained using the FS:[30] register expression. The Init function is resolved from exports of botdll injected into services.exe by a second APC (APC2), as described later. APC1 also creates a new thread in user mode, whose startup routine is shown below:

```
ThreadStartRoutine:
push    1
push    0FFFFFFFFh
call    dword ptr [esp+0Ch]
                ; SleepEx(INFINITE, TRUE)
jmp     ThreadStartRoutine
```

This thread seems to be doing nothing but sleep forever! The purpose of this sleep is to put the thread in an alertable state using the bAlertable parameter as TRUE so that future APCs can be executed promptly:

```
DWORD SleepEx(
    DWORD dwMilliseconds,
    BOOL bAlertable
);
```

If the thread is not in an alertable state, APCs are queued [7].

Setting up botdll: step 2

The next step in setting up the user-mode botdll is for the malware to read its own driver file from disk. It first creates an empty file object using ObCreateObject and sets the file

name to refer to its own driver file. It then gets the device object of the lowest file system driver, i.e. NTFS driver, and, using the new file object and device object, generates IRP_MJ_CREATE to read its own driver file.

The file is read in two steps. First, the file size is obtained using IRP_MJ_QUERY_INFORMATION with FILE_INFORMATION_CLASS as FileStandardInformation. In the second step, IRP_MJ_READ is sent in a buffer allocated from ExAllocatePoolWithTag. Rustock.C then sends IRP_MJ_CLEANUP and IRP_MJ_CLOSE directly to the NTFS driver to undo the actions associated with IRP_MJ_CREATE. The memory location containing the malware's own file data is saved for later use (for example, in a separate worker thread to write its copy to disk at regular intervals for resuscitation).

Typically, IRP_MJ_CREATE, IRP_MJ_CLEANUP and IRP_MJ_CLOSE are generated implicitly by I/O Manager inside the *Windows* kernel, and by rolling out these IRPs on its own, Rustock.C showcases the sophistication of its authors. Generating its own IRP_MJ_CREATE is a non-trivial task involving several intricate steps, especially relating to setting parameters for the caller's security context. Since it rolls out its own IRP_MJ_CREATE, the Rustock.C driver is able to send *direct* read and write requests (IRP_MJ_READ and IRP_MJ_WRITE) to the NTFS driver. This allows the malware to bypass any filter drivers that are typically used by security vendors to provide kernel-based on-access security against malicious files.

From the data buffer containing the on-disk driver the next step is to get botdll. The botdll code is stored encrypted and compressed in the original driver file, as shown in Figure 2.

The encryption consists of a simple XOR and the compression algorithm used is aPLib [1]. After the botdll code is uncompressed into a new memory buffer, it is virtually mapped into yet another new buffer. Relocations of the botdll code are fixed in kernel mode as its user-mode base address has already been obtained from the MmMapLockedPages call. PE and MZ signatures in the final virtually mapped buffer are also zeroed out and the start address of that buffer is set in the NormalContext field of the second APC (APC2). Like APC1, APC2 is also queued to threads of the services.exe process. The NormalRoutine parameter of APC2 consists of code that performs fixups of imports of botdll. The imports are fixed up using the LoadLibraryA and GetProcAddress APIs that have already been set up via APC1.

An anti-emulation/anti-debugging trick is used for import table fixups in botdll. First, it looks up the virtual address of byte sequence c20400 in the kernel32!SetEvent function, which is actually the machine code equivalent of mnemonic 'ret 4':

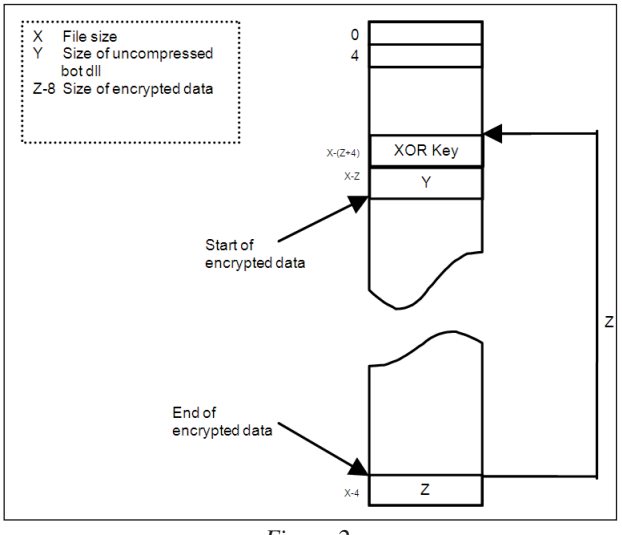


Figure 2.

```
00011093 call   loc_00011098
00011098 pop    edx
00011099 add    edx, 10h ; Save 000110A8 in EDX.
0001109C push   0      ; Extra push1.
                       ; PUSH to compensate for
                       ; additional sizeof(dword)=4
                       ; byte increment in esp,
                       ; because of 'ret 4'.
0001109E push   edx    ; Extra push2.
                       ; Pushing location 000110A8.
                       ; Location 000110A8 is where
                       ; ret 4 instruction
                       ; will transfer control.
0001109F push   [ebp+Va_ImpDllName]
                       ; e.g. "kernel32.dll".
000110A2 push   [ebp+Va_RET_4]    ; Extra push3.
                       ; Pushing address of location
                       ; in kernel32!SetEvent whose
                       ; opcode is ret 4.
000110A5 jmp    [ebp+Va_LoadLibraryA]
                       ; This is where the return call
                       ; trick is executed.
000110A8 mov    [ebp-38h], eax
                       ; Save return from LoadLibraryA
```

The calls to GetProcAddress are made in a similar way, by using the address of the 'ret 4' instruction taken from the kernel32!SetEvent function. After relocations, the entry point of botdll is called, followed by a call to its export function, named Init, that performs the bot activity.

ACTIVITIES OF KERNEL THREAD2

Thread2 is created from Thread1 and writes its own driver file to disk every five seconds in a loop. This is most likely its persistence strategy against any deletions of its on-disk

driver file. Its own driver file is saved in memory during the startup phase. Similar to the reading of its driver file, it performs its write (IRP_MJ_WRITE) by direct access to the NTFS driver, bypassing the file system filter device stack.

ACTIVITIES OF THREAD3

Thread3 is created conditionally from the process create notify routine. This thread does the same work as that carried out towards the end of Thread1, which involves reading its own driver, sending APC1, decompressing botdll and sending APC2. In the notify routine it checks for process create only notifications of services.exe. The condition to create a new thread is whether the botdll code has been spawned into services.exe previously or not. Most likely Thread3 is employed as a backup mechanism to kick off APC1 and APC2, since there may be a race condition in the boot phase between the driver's startup and the startup of services.exe. If, by the time the driver's startup has completed services.exe has not started, then Thread3 can kick off APC1 and APC2.

DISPATCH ROUTINES

The Rustock.C driver has no dispatch routines set up in its DRIVER_OBJECT, as there would be in the DriverEntry routine of a typical device driver. However, it accomplishes a similar objective using an array of 11 functions set up in memory. For example:

- Dispatch function 0 frees up the current driver in memory, reads its own disk driver afresh and subsequently sends APC1 and APC2 as described earlier.
- Dispatch function 1 writes a new driver using IRP_MJ_WRITE. This can potentially be used to activate a completely new driver downloaded from botdll.
- Dispatch function 2 deletes a disk file, using IRP_MJ_SET_INFORMATION and FileInformationClass as FileDispositionInformation.

All disk access in these dispatch functions is also achieved via direct calls to the NTFS driver as described earlier. Each of these functions is called through the hooked ZwTerminateProcess API, by setting a function index along with the corresponding input/output parameters. The layout of the input/output structure is described below:

```
struct ZwTermProcDispatchIOParam
{
+0x0   FunctionIndex    // Index into function array
+0x4   InputBuffer      // Input buffer, if applicable
+0x8   InputBufferSize  // Input buffer size, if
applicable
+0xC   OutputBuffer     // Output buffer, if applicable
+0x10  OutputBufferSize // Output buffer size, if
applicable
}
```

The address of this structure is passed in as the first parameter to the ZwTerminateProcess API and the second parameter (ExitCode) consists of the special encoded value as described earlier.

REMOVAL

The Rustock.C variant researched in this paper was removed by deleting its driver service registry keys under HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\<Rustock_Driver_Name>, followed by a system reboot and deletion of its driver file. But the registry hiding mechanism it employs needs to be defeated before access can be gained to its driver service registry keys.

CONCLUSION

Rustock.C has the ability to operate with its bare minimum driver file containing another driver file and botdll, both stored compressed and encrypted. The mapping of its own driver image in the context of a user-mode goat process, combined with the use of the APC mechanism, obviates the need to have the botdll on disk. Some of the dispatch functions implemented via the ZwTerminateProcess hook demonstrate its ability to activate a completely different botdll and driver on the fly. Any access (read, write, query or set information) to its own driver file on disk is done surreptitiously, bypassing the file system filter device stack.

REFERENCES

[1] Kwiatek, L.; Litawa, S. Yet another Rustock analysis... Virus Bulletin, August 2008, p.4.

[2] Kwiatek, L. Rustock.C – kernel mode protector. http://www.eset.com/threat-center/blog/?p=127.

[3] IA32_SYSENTER_EIP. http://uninformed.org/index.cgi?v=8&a=2&p=12.

[4] Stealth Rootkit Designed for Vista. http://news.softpedia.com/news/Stealth-Rootkit-Designed-for-Vista-30108.shtml.

[5] Skape & Skywing, A Catalog of Local Windows Kernel-mode Backdoor Techniques. http://www.uninformed.org/?v=8&a=2&t=txt.

[6] Bypassing Windows hardware-enforced data execution prevention. http://www.uninformed.org/?v=2&a=4.

[7] Asynchronous Procedure Calls. http://msdn.microsoft.com/en-us/library/ms681951.aspx.

OPINION

FAMILY MATTERS

Henk van Roest
Microsoft, UK

Imagine the scenario: I am just sitting down to dinner and the phone rings – great timing! The caller ID reveals that it is my sister. She has a question: my niece and nephew are going to be using PCs for school and homework. How does she secure the home network and PCs? How does she, as a concerned parent, filter the Internet content she feels is inappropriate for her children? Initially, it seems this is going to be a short call, but as the conversation unfolds, the uncertainties and concerns mount up and any hopes of being able to eat my dinner before it gets cold are dashed.

While searching the Internet for anti-virus and parental filter products, my sister had been confronted with a plethora of results. Knowing just how careful she has to be before clicking on a random link, she decided it was best to ring her brother. What do people *without* any tech-savvy relatives do, I wonder? Whom do they ring?

THE EROSION OF TRUST

Recently, another family member was presented with a very professional-looking web page and a pop-up message alerting him to the presence of malware on his machine. For only a small payment, suggested the pop-up, he could download an application that would both remove the threat and protect the machine in the future. Needless to say, I spent some hours the following weekend removing the application, which exists somewhere on the border between legitimate and illegal software.

The Internet is a wonderful resource. However, it also presents the criminal fraternity with the means to exploit others in ways the old-time crooks could only dream of. While this is not new or even news, to the *non-technical* user, it presents a real dilemma. It is no longer safe to surf the web, click on links and see where the journey takes you. With cybercrime an ever-growing problem, one is faced with the question: where do I go for trusted, impartial advice?

There is no doubt that competition and choice greatly benefit the consumer. As an industry, we must continue to innovate, drive each other to new heights of functionality and protect all of our shared consumers. However, the wide choice of legitimate products and services, together with the criminal options available on the Internet creates doubt. A 2006 survey suggested that, in the UK, citizens fear online fraud and criminal activity more than physical crime (http://www.getsafeonline.org/nqcontent.cfm?a_id=1438); I doubt that this statistic is limited to the UK.

TIME TO TAKE CONTROL?

I am not an advocate of Internet censorship; the right to free speech should be defended with all the courage one can muster.

The Internet should be a place where all are free to explore, learn, be entertained and do business safely. When my niece and nephew go to the local library, I don't expect them to be targeted by muggers, yet somehow we do not display nearly the same outrage when this happens on the Internet every day.

If your home was invaded and your possessions stolen, it would no doubt be a traumatic experience; yet every day our PCs are subjected to virtual invasions and theft. A recent survey in the UK discovered that owners would be willing to pay up to £5,000 for the recovery of the information on their PC should it be stolen.

As a group of security professionals, enterprises and governments, we must protect our citizens against the increasingly organized criminal element trying to take control of one of the world's most important resources.

Each country has its own laws and standards of acceptable behaviour, yet this should not stop us from agreeing to a lowest common denominator. As interested parties in the IT security field, we need to coordinate our efforts for the sake of all our customers. When a customer connects to their Internet banking site without realizing their PC has been hijacked, the money stolen from their account may be their own (or in some cases the bank's), but the real cost of the theft affects us all.

Some questions of fraud come to mind:

- Are online businesses being defrauded by the next online order using stolen credit card details?
- Is it acceptable for a customer of a financial institution to be denied access to their online account information because their PC is inadequately protected?
- Should an ISP be able to deny a paying customer access to the Internet simply because they lack adequate software and security protection?
- Is it acceptable to prey on the doubts and fears of our citizens by offering them insurance against identity theft? Is as much effort put into preventing the theft as into convincing people to invest in insurance after the fact?

In order to counter the threats present in today's Internet environment, we have to think differently. Maintaining civil liberties while protecting the majority will be difficult. I don't think anyone would advocate the introduction of an Internet 'police state', but should there be a PC 'driving licence'?

A MOVE TO A CURE?

'*Our problems are man-made; therefore they may be solved by man. No problem of human destiny is beyond human beings.*' John F. Kennedy.

In legal terms, Internet crime is relatively new; laws across different countries are inconsistent and the global nature of the Internet allows for crimes to be perpetrated anonymously and remotely with little fear of prosecution.

If there is a will, then as a global community, we can fight this cancer that is affecting our ability to conduct business globally in a safe and secure environment.

Is it possible for software, hardware and service vendors of the world to put aside their differences and form a powerful alliance with government and law enforcement to wage war against the online criminal element?

Whilst the value of online business has been growing year on year, many consumers still fear fraud and crime online. Driving the criminal element from the Internet would allow this market to flourish even more to the benefit of all businesses, shareholders and consumers.

Is it time for a United Nations of the Internet? Rather than individual nations working in isolation, can an institution recognized for its mediation in worldwide affairs bring the force of its members into the online world?

THE RE-ESTABLISHMENT OF TRUST

If my home is invaded, I expect the police to investigate and bring the perpetrators to justice. No such recourse exists online. Our homes are constantly under attack from remote and anonymous sources and we potentially invite the perpetrators into our homes while surfing the Internet.

If we can instil a sense of responsibility into people to learn about the operation of their PCs in the same way as they learn to drive a car and recognize the hazards on the road, perhaps we can prevent online crime more effectively.

In the UK, government and private enterprise have joined forces to help educate and provide proactive Internet safety guidance to its citizens through the Get Safe Online initiative. Millions of people have benefited from the campaign, either through its online presence, television advertising or workshops held across the country in shopping centres and educational establishments.

While initiatives like these are to be welcomed as a step towards bringing a trusted advisor back into the online lives of the citizenship, one cannot help but ask: is education enough? Is awareness ever going to re-establish trust or should we contemplate the cure and perhaps give up some of our freedom online?

CONFERENCE REPORT

THE OTTAWA RULES

Helen Martin

This year the *VB* conference returned to Canada and paid a visit to Ottawa, Canada's fourth largest city and seat of the country's federal government. The venue for this year's conference, the Westin hotel, couldn't have been in a more convenient position for exploring the city – with the Rideau canal within spitting distance, Parliament Hill and Byward Market a couple of steps away, and museums of war, nature, contemporary photography and the Royal Canadian Mint within less than a mile. But city explorations were put on hold – for three days at least – as the doors opened on the 18th Virus Bulletin conference.

IT'S ALL GEEK

The conference kicked off on Wednesday morning with *Sunbelt* CEO and blogger extraordinaire Alex Eckelberry taking to the stage after the official conference opening for his keynote address: 'The AV industry – quo vadis?'. Alex compared statistics from two surveys – one of consumers and another of enterprise customers – that explored customers' feelings towards their AV products and vendors. Overall, consumers appeared to be more satisfied with their products and to place greater trust in their vendors than enterprise users. Alex also stressed the importance of customer support in gaining consumer confidence and presented the results of a review of vendor technical support services – showing many to be lacking in various areas. A perfect start to the conference, the address was entertaining, engaging and struck a chord with pretty much everyone in the room.

After the keynote address the conference split into its usual two-stream format, with David Emm presenting an overview of the malware business, or 'the flip side of the legitimate economy', in the corporate stream, while in the technical stream Morton Swimmer posed the questions: 'How can we build an effective defence structure?', 'How can we get our products to work *together*?' and 'What models can we use for product interaction?'.

After lunch, Gunter Ollmann and Holly Stewart covered the merging of the underground markets dealing in malware and vulnerability exploits, and discussed how the competitiveness between the different market areas actually makes it easier for security vendors to detect the threats.

Kimmo Kasslin's presentation proved to be the most popular session of the conference, taking a detailed look at Mebroot, one of the most sophisticated pieces

Jeannette Jarvis shows the Ottawa Gee-Gees how it's done.

of malware seen in recent times, and characterizing it as 'commercial-grade framework'.

In his paper Matt McCormack coupled analysis of the major malware families targeted by the *Microsoft Malicious Software Removal Tool* with the telemetry it gathers, to provide a perspective on how malware authors respond to the impact on their networks after each release of the disinfection tool. Matt's observations indicate that being targeted by the tool causes significant changes in malware behaviour, including increased use of evasion and stealth techniques.

Jeff Aboud viewed the malware problem from a different angle – that of anti-malware marketing. The mass outbreaks of the 1990s created what seemed like ideal marketing conditions – a situation in which the media and prospective customers all wanted to hear what the anti-malware companies had to say. With the lack of big outbreaks in recent years Jeff argued that many vendors have found a gap in their marketing strategy. He discussed a 'threat marketing' strategy, describing how it can be implemented to help vendors keep their name in front of prospects and key stakeholders.

Following the last of the day's scheduled presentations, *ESET*'s David Harley took to the stage for the company's sponsor presentation, 'Interpreting threat data from the cloud', after which it was time to head for the bar.

Wednesday evening saw the first of the main networking events of the conference – the VB2008 welcome drinks reception. Delegates were welcomed at the door by two burly hockey players from the University of Ottawa team the Gee-Gees, providing some excellent photo opportunities and the chance for delegates to practise their bully-offs.

It was also at the drinks reception that a certain piece of distinguished head gear began its magical mystery tour – spot the real owner!

LOGIC BOMB

Day two started bright and early at 9am with Ismael Briones describing an automated classification system that uses graph theory to identify malicious files with similar internal structures. Meanwhile, Gunter Ollmann took to the stage for the second day running, this time stepping in for a colleague who was unavoidably detained and presenting an interesting paper on the security of virtualized networks.

After mid-morning coffee the schedule in the technical stream turned to anti-spam, more on which later. Meanwhile, David Perry held court in the corporate stream with a paper charting the life and death of the pattern file, followed by Oliver Auerbach, who described how *Avira* handles the never-ending flood of malicious file submissions using a tool which handles deduplication and assigns tasks to analysts according to priority and relevance.

Thursday afternoon saw the return of the last-minute technical presentations following their success at last year's conference. The proposals for these shorter-format (20-minute) presentations were submitted and selected just three weeks prior to the conference, allowing for subjects that were more up-to-the-minute than the full length papers. The fast-paced 'turbo' talks were started off by *VB*'s head of testing John Hawes, who outlined details of a new anti-malware testing methodology that *VB* is planning to introduce to supplement the information provided in the VB100 comparative reviews. Boris Lau followed, with a look at how malware authors effectively emulate the 'race to zero' contests held by other security events as they attempt to beat online scanners as a matter of course.

Spot the rightful owner!

John Hawes talks RAP testing.

Next up, Pedro Bueno provided an insight into the world of South American cybercriminals and their banking trojans, followed by Marius van Oers with a look at what can be done on the *Apple iPhone* with an SDK, and what possible new malware attack vectors could arise from it.

After a quick break for tea, Dan Hubbard presented a different take on the technology *du jour*, cloud computing, discussing how it can be used to decentralize attacks and how it opens up new opportunities and threats to security researchers. Kurt Baumgartner then presented an overview of recent rogueware, which was followed by Sorin Mustaca's presentation in which he introduced an aggregator for phishing and other malicious URLs. The final last-minute presentation was given by Nicolas Brulez, who gave a live demonstration of a malicious packer, showing how it is possible to manipulate unpacking routines.


Panel members share their views on testing.

To round off day two's presentation schedule delegates gathered in the technical stream for a panel discussion entitled 'The current state of anti-malware testing'. Led by Stuart Taylor, panel members representing an end-user, the media, testing bodies and a vendor (John Alexander of *Lockheed Martin*, Paul Roberts of the *451 Group*, Andreas Marx of *AV-Test* and *VB*'s John Hawes, and Righard Zwienenberg of *Norman*, respectively) answered questions from the floor ranging from whether the panel felt that testing is stifling innovation in detection technologies, to how panel members foresee the testing of products that use in-the-cloud technologies. As is often the case with panel discussions, the 40 minutes flew past leaving much unsaid – an indication of how much interest currently surrounds the topic of anti-malware testing.

SPAMALOT

Five years since the introduction of the *VB Spam Supplement* and four years since spam-related papers were first presented at the *VB* conference, spam continues to clutter up our inboxes and shows no sign of abating. This year's conference included five papers on the subject. On the technical side, Patrik Ostrihon and Reza Rajabiun looked at the robustness of new email authentication standards, and Andrey Bakhmutov described a method for tracking botnets sending out spam, while in the corporate stream Vipul Sharma presented a case study of non-English spam, Darya Gudkova described a view of Russian spammers and Chris Lewis reported on *Nortel*'s open-source spam filter for enterprises. A sixth spam paper was scheduled, but following the non-appearance of both the scheduled speaker and the reserve speaker, it was Martin Overton who gallantly stepped up to the mark with his paper on malware forensics. Our heartfelt thanks go to Martin for saving the day.

DATA DIDDLER

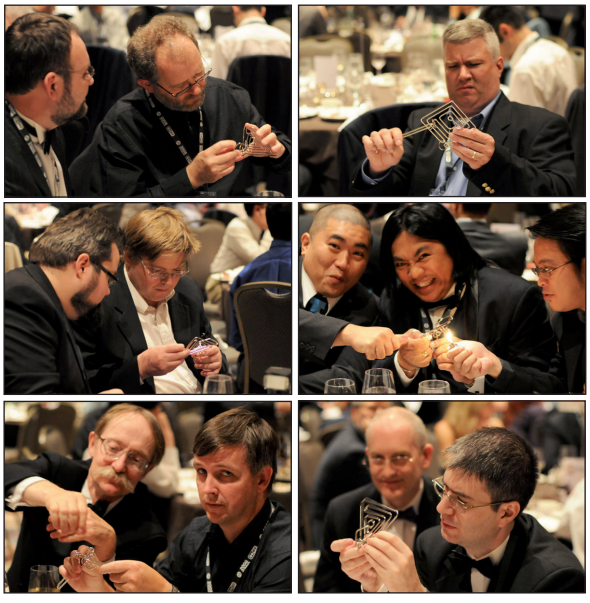

Hosts of the VB100 quiz, Graham Cluley and yours truly.

Thursday evening was, of course, gala dinner night – or was it quiz night? This year's gala entertainment was somewhat more interactive than normal – there was to be no sitting back and waiting to be entertained this year! Compèred by myself and Graham Cluley, the idea of the evening was for dinner tables to compete against each other in a battle of wit and trivia with a selection of exciting prizes on offer for the winning team.


Alex Shipp demonstrates he still pays heed to the Scouts' motto 'be prepared'.

Karen Richardson proved she was truly game for a laugh as she took to the stage in character as the joker, geeing up the audience with 'ooh's and 'aah's and being as flabbergasted as the rest of us when, on being sent on an errand with Alex Shipp, her running partner stripped down to running shorts and trainers in the middle of the dining room.

After five tough rounds of questions based loosely on the subject of malware, ranging from geek trivia to mind-bending brain teasers, just when the audience thought it was safe to sit back and relax, it was time for the final challenge. Two Chinese-style wire puzzles were given to each team, the challenge being to complete both puzzles before the final scores had been totted up. The photographs on the next page give some indication of the level of frustration caused by the 'mosquito' and 'gridlock', and congratulations go to Nick FitzGerald for being the first to complete both puzzles. Nick (and others who eventually completed the puzzles) will be pleased to know that,



'Brain power alone is the key to success.'

according the puzzles' manufacturer, 'brain power alone is the key to success'.

The eventual winners of the quiz were team 'Chop Chop', closely followed by runners-up 'The WTFs' and 'K9s'. A mention should also go to the losing team, 'Rødgrød med Flød' – who were 'rewarded' for their efforts with a very special booby prize.

Team 'Rødgrød med Flød' seemed happy with the booby prize.

Special thanks go to Graham Cluley, whose wit sparkled as brilliantly as his fuchsia pink accessories and whose skills as a quiz show host will hold him in good stead for the day the Eurovision Song Contest returns to the UK, as well as to the members of the *Cue Media* team who put such an enormous amount of work into the production of the show's graphics and its staging.

DISASSEMBLY

For those whose brains had recovered sufficiently from the previous night's mental workout, the final day of the conference began at 9.40am. In the corporate stream Randy Abrams described how a household appliance can be used as a means to teach users about bots and botnets, while in the technical stream Andrew Walenstein and Arun Lakhotia demonstrated the use of game theory to assess the strength of an AV system against evolving offences.

Two papers on anti-malware testing followed the mid-morning coffee break, with Andrew Lee questioning whether it is possible to make testers and certifying authorities more accountable for the quality of their testing methods and the accuracy of the conclusions they draw, and Igor Muttik looking at rebuilding anti-malware testing for the future.

Other highlights on Friday included Ryan Hicks' overview of rules-based analysis using *IDA Pro* and CLIPS, Richard Ford's outline of a new automated sample submission/multi-scanner service, and Peter Ször's paper exploring the possibility of malware evolving to follow Darwinian principles – while still very theoretical, the paper provided plenty of food for thought.

Rounding off the conference was a discussion forum on security in banking – this despite the fact that during the week leading up to the conference it seemed doubtful as to whether there would be any banks left to worry about security. Session chair Jan Hruska directed questions to independent researcher Nick FitzGerald, Reza Rajabiun of York University and *COMDOM Software*, and Eric Davis from *Google*. The discussion opened with a question to the audience: 'Will there be a change in phishing volumes due to the current global banking crisis?' Opinion was somewhat divided, although we now know that the rate of phishing has indeed increased over recent weeks. The discussion moved on to cover liability for phishing losses and user education. The panel session ended while there was still a sea of raised hands in the audience – but was concluded, suitably enough, with the comment 'Users are stupid and will remain stupid', from whom else but Vesselin Bontchev.

AND FINALLY...

My thanks to the VB team, Karen Richardson, the Cue Media team and students from Carleton University.

There has not been enough space to mention more than a small selection of the speakers and presentations here, but I would like to extend my warmest thanks to all of the VB2008 speakers for their contributions, as well as to sponsors *ESET*, *ParetoLogic*, *COMDOM Software*, *OPSWAT*, *TrustPort*, *Sunbelt Software* and *K7 Computing* for their support.

Next year the VB conference lands on the shores of Lake Geneva, with the conference taking place 23–25 September 2009 at the Crowne Plaza, Geneva, Switzerland. I very much look forward to welcoming you all there.

Photographs courtesy of: Andreas Marx, Petr Odehnal, Jeannette Jarvis, Kenneth Bechtel, Joe Wells, Tjark Auerbach and Marius van Oers. For more photographs see http://www.virusbtn.com/conference/vb2008/photos.

PRODUCT REVIEW

DRIVESENTRY DESKTOP 3.1/3.2 & GOANYWHERE 1.0.2/2.0

John Hawes

This month marks a bit of a departure from the norm, as we look at one of a growing 'new breed' of security products, which focuses less on the traditional arts of the anti-malware world, such as detection of malware via signatures, heuristics and behaviour patterns, and aims instead to protect systems by preventing unauthorized software from performing any potentially dangerous activity.

DriveSentry is a fairly young company, set up in 2005, but has managed to generate quite some buzz around its product line. This has been helped, no doubt, by the company giving away the basic versions of its software and making money on upgrades to slicker, higher-spec editions – an increasingly common practice among security vendors. The product range is pretty basic, with the *DriveSentry* technology available as either a standard desktop product or as a special version designed to protect the growing range of removal and portable storage devices on the market. I took a quick look at both tools to see how they fared in the *VB* test lab.

WEB PRESENCE, INFORMATION AND SUPPORT

DriveSentry's website (www.drivesentry.com) is a pretty groovy place, presenting a cool black background with lots of funky graphics and animation. The home page heavily promotes the company's flagship products, proclaiming them to be 'next-generation anti-virus', with a graphic showing various types of drives and storage devices being protected by a nimble line of defence. Fat shiny buttons provide access to product downloads, feedback areas and a recommendation system, encouraging users who have been impressed by the product to pass it on to their friends. Much of the site, and indeed the products, takes a similarly 'Web 2.0'-approach, with lots of user-generated content and interactivity.

Various help and support buttons seem only to lead to a range of nifty little videos guiding users through the various steps of installation and setup, with screen recordings slowly following the prescribed path through the product to perform various tasks. The company's logo, a heavily armed, red-jacketed guardsman standing watch over an open hard drive, adorns the intros. The videos provide a fairly extensive guide to the operation of the product, in a very simple, unwordy manner that would be accessible to the most uneducated and unskilled computer user, but may be a little slow for those with a short attention span. More detailed assistance is provided via a forum, traffic on which currently seems to be mostly company-generated, but which clearly has a select core of regulars posting both problems and advice to other users. Responses from company reps are refreshingly open, polite and helpful as well as prompt. There are also fairly well-stocked threads about the products on the Wilders Security forum and others elsewhere, again with good input from the company.

For users with more urgent problems, *DriveSentry* also provides an online chat system, which of course merited a brief trial. A contact greeted me almost immediately, and a few deliberately vague queries about some issues with the product were answered in a similarly prompt, friendly and helpful manner.

The website also provides some more in-depth information on the product and its sales model. The desktop version is provided free of charge, but with a one-off licensing fee payable for continued access to automated 'tricklefeed' updating after the first month – non-paying users can continue to update manually. The protection system is described as 'tri-security', referring to the blacklist of known malware (variously counted on the site as containing 'over 1 million', 'over 1.3 million', and '1,866,852' unique items) and a whitelist of trusted software (with a much smaller content: fewer than 50,000 items) operating alongside a herd-immunity system which relies on the input of other users to decide whether to trust an item.

Elsewhere the site offers information about the company, a news section featuring both company news and feeds from some major security news sources, and on the home page a scrolling 'threats' section promises a range of interesting-looking articles on topics including threat analysis and general security advice. A final adornment to the home page is the *Check Mark* logo, indicating that the company's products have achieved *Check Mark* certified status. Further investigation revealed that the products had managed complete coverage of the January 2008 WildList, which is an impressive feat for such a young and apparently small firm given the range of complex polymorphic viruses in the list.

With this factoid to whet my appetite, I headed into the lab with a freshly downloaded copy of the software, and had a look for myself.

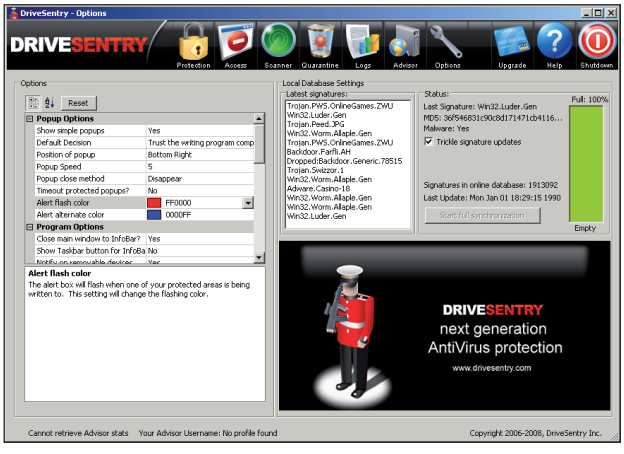
INSTALLATION AND CONFIGURATION

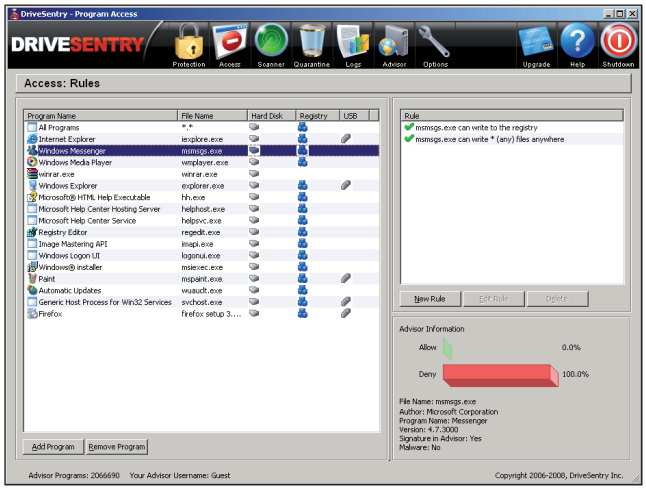
Initial setup of the product is simplicity itself, with a small download of 30 MB or so and an installation process running through the standard pattern of introduction, selection of install location, EULA, file copying and release notes. Once this is complete the product has to download its white- and blacklist data. This takes a little longer (around ten minutes in some cases), and at the first attempt there

was a problem after reaching 99% of the download – an issue apparently caused by the remote copy of the database having been updated before copying could complete. This was fixed in the next version of the product, which was available later the same day. With the databases fully stocked, a little tour of the interface was in order. It proved pretty simple to navigate, clear and slick.

The main pages show the range of folders, file types and registry keys being monitored by the software. The file types and registry key settings are pretty exhaustive, with most of the important areas covered, but the default folder list seems a little sparse, limited to the personal documents areas and the folder where the *Windows* hosts file is kept. All of these lists are simple to modify, with additions and changes to existing settings able to be put in place within a few clicks, but some more advice on the potential impact of such changes might be appropriate in some cases. More savvy users might, for example, want to keep an eye on other areas – such as the system32 folder, a pretty standard location for malware to use. This can be done fairly easily, but should be expected to generate considerably more prompt messages. Further tabs under the same section provide details of the default actions for various items and areas under protection. The default is to 'prompt' in most cases, and there is a list of trusted applications installed on the current system, a sensible method which avoids the exhaustive but often unnecessary lists of trusted apps presented by some products with similar functionality – software installed to the system at any time is added to the list as appropriate.

The 'access' screen lists the trusted programs again, with some more information on each. Each is accompanied by details of the level of access afforded to the software in question, including the ability to write to disk, to change registry settings, and access to USB storage devices. Here also are the community-rating scores for each program, which seemed to vary wildly from 100% in favour of





trusting to utterly against – somewhat surprising for a clean system carrying almost entirely software included as part of a *Windows* installation.

The next few tabs provide the on-demand scanner, along with the quarantine, from where files detected as malicious can easily be restored or deleted, and logging, which provides details of the product's various monitoring and blocking activities. A tab labelled 'advisor' links in with the company's herd-immunity scheme, and presents some lovely maps of the locations of the users' connections to the system, along with the latest threats and newly trusted files identified around the world.

An 'options' button provides a good range of fine-tuning choices, from the detail level and even colours of the alert/prompt popups, to default behaviours such as automatically allowing whitelisted items. This 'auto-allow' function is available only for the first 30 days of use and to paying users (a single, one-off fee) thereafter, but also extends to the community scheme; automatic responses can be further fine-tuned to trust the opinions of the group, both in terms of the percentage trusting or blocking a given item and also the actual number of opinions registered, with a default of 90% of 20 users.

A final set of buttons provide access to the upgrade facility, allowing users to pay for ongoing access to the automated updating and responses to new items. A 'Help' button is also provided, which opens a reasonable HTML guide hosted on the company website. The guide covers most of the product's functionality in simple, clear terms, focusing on a page-by-page run-through of the options available and their purpose. A large shutdown button allows users to minimize the interface, switch off all functionality while remaining active and ready for reactivation, or shut down the product completely.

DriveSentry seems to be in a constant cycle of bug fixing and new builds, and several versions I tried over a few

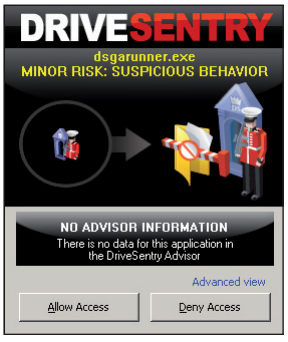
weeks of testing had minor issues. Beyond the update problem mentioned earlier, one version decided it could not access the Internet for the Advisor data, despite having quite happily stocked up its databases just minutes earlier. This meant the machine ran at a snail's pace as each attempt to perform any action was met with an attempt to retrieve inaccessible data before deeming it permissible. The active community and speedy release cycle means that most issues are fixed within days if not hours, but generally a reinstall seems to be required.

SYSTEM PROTECTION

Having familiarized myself thoroughly with the layout of the interface, it was time to have a look at the protection capabilities of the product.

I started off with the simple on-demand scanner. This is fairly clearly laid out, with a list of options for scan types including the default full scan of all drives, a core scan of known sensitive areas, and various other settings including specific files or folders. This was run over a few sets of malicious items, and I was pleasantly surprised with the product's coverage of the WildList and other sets of recent malware, with the static items, trojans, worms and so on obviously better covered than the more esoteric file-infecting viruses. From the test set used in the most recent VB100 certification review, around 85% of static samples were detected on demand, but less than 1% of file-infecting viruses were detected. On-access scanning was also in place, with the same known malcode detected even when being copied to unmonitored folders by trusted applications. This side of the detection system is clearly based on simple hashing rather than in-depth file analysis, and is thus only capable of detecting files seen by the company's lab. However, it remains fairly speedy and shows that some sterling work is being done by the lab in both gathering and processing samples. For what is essentially an unexpected extra to what I had assumed would be a straight IDS/whitelisting product, it performed impressively well.

Moving on to some more exacting tests, the system was disconnected from the web for safety and some infectious samples were executed to see how they were handled. Again things were pretty impressive, with the vast majority of malicious items quickly bringing up an alert box asking if disk or registry access should be allowed to an unknown process. This method is not ideal though, automatic blocking of malicious activities always being preferable to putting the onus of decision on the user, but in many cases the mere hindrance to their running caused the more cautious malware to stop running, and most were easily deactivated after clicking the 'block' button. In the hands of less cautious or less experienced users, the system may present difficulties, particularly when the popup presents one of the more legitimate-sounding filenames used by some malware, which could easily convince the unwary that some normal everyday activity was going on.



Of course, with such a simple method of protection a few items were bound to slip through – in many instances worms and data-stealers were allowed to perform pretty comprehensive trawls through the system looking for email addresses, passwords etc., performing all their misdeeds without writing to disk and thus without coming to the product's attention. In a couple of instances more surprising behaviours were discovered, including at one point finding the desktop flipped to what appeared to be the login page of a Brazilian bank, while a nasty fake anti-malware product managed to present a spoof blue-screen, riddled with warnings of malware infection, followed by a mock-up of the *XP* boot process, advising that some no-name product should be purchased to clean up a dangerously infected system. However, in this case at least the malware was stopped in its tracks before its final stage, as disk writing became involved. Finally, an autorun worm was allowed not only to drop a copy of itself to a USB storage device attached to the system, but also to configure itself to autorun on insertion into a new system.

Moving on to clean items, no false positives were in evidence when the product was run over the *VB* clean test sets, and installation of a range of software generally presented informative and usable requests for permission to carry out various tasks, always with the option simply to add a new piece of software to the trusted list. However, in many cases there seemed to be a bit of a deficiency in the whitelisting side of things, with items which one would assume would have had pretty decent penetration in a range of user-bases apparently not covered by the central system. In such cases the user is left to rely on his own judgement and that of his fellow users as to whether it is safe to install and trust the applications.

OTHER FUNCTIONALITY

Having learnt a fair amount about the desktop product, it was time to move on to its intriguing little sister, *GoAnywhere*. This promised complete and portable protection for USB and other types of storage device, along with tiny system impact. The download and setup process was fairly straightforward, with a much smaller file to download and the process of copying to a given device

quick and easy. A quick inspection showed that the product is indeed simple in the extreme – essentially a pared-down version of the desktop product, with a small folder of files taking up only a few MB on the target device. The product uses autorun functionality to launch itself, contact the online databases of trust, and monitor for unwanted data attempting to write to the drive.

This immediately rang a few alarm bells, as this kind of autorun mechanism is a clear security risk and something we generally advise people to disable whenever possible, particularly on USB drives as so many worms now use it to spread from system to system. However, in this context it is perhaps a valid technique, as it means that drives can be inserted with a greater degree of security into unsecured machines such as cybercafé systems, which are likely both to have autorun enabled and to be carriers of malware.

Installation highlighted some of the shortcomings in the whitelisting side of the main product, which prompted for opinions on various parts of the *GoAnywhere* installer and file-copying process. Once up and running, the operation and configuration is fairly straightforward, offering some general system protection as well as that on the removable device. Quite a few popups prompted for choices on connecting to a new machine – no configuration is transferred across, so numerous prompts appear even for processes like *Explorer* doing its routine actions, and one clumsy mis-click led to it being shut down, which didn't help the machine much. Lacking a web connection led to some unhappiness too, as the product slowed the system down to a crawl when unable to find its data.

A new set of features including 'powerful' (AES 256-bit) drag-and-drop encryption is promised in a forthcoming upgrade to the product. A quick look at a pre-release version of this showed it to be buggy, however, with numerous problems including a failure to launch at all on transferring encrypted data back to the original system, leaving the secured files inaccessible. The system is pleasantly designed though, and should be fairly usable once fully ready for action.

CONCLUSIONS

There is a little confusion about exactly what market *DriveSentry* is aimed at. During the installation process the standard messages about removing any conflicting security software are in evidence, but from personal communication with the company and various postings and discussions on forums it seems that the product is designed to be compatible with more traditional anti-malware solutions, intended as an extra layer of security in addition to, rather than in place of, these more standard products. For this purpose it seems like an ingenious, simple tool with some excellent protection capabilities. There are, as with every product, a few holes which doubtless could be exploited should the user be unlucky enough to be hit by exactly the wrong piece of malware, but this remains a danger with even the most sophisticated and complex security setup.

There are a few areas which could perhaps be beefed up a little, including the products' whitelists of trusted files. Supporting this effort with a level of community involvement is an interesting concept, one which has been rolled into a variety of products of late, and which does seem to provide some benefits. However, for the more cynical the 'ask the audience' approach will inevitably bring to mind the legions of 'instant security experts' encountered in the security space, each with a unique and often bizarre slant on security issues and how best to resolve them. Just how far the opinions of the world at large can be trusted is difficult to judge, but the provision of fine-tuning controls does at least allow users to decide how much faith to put in the community's opinions, and many will simply take account of the majority opinions when making decisions.

This, of course, opens up another can of worms: that of how far users should trust themselves to make decisions about what software to trust. Prompting for permission, where used in operating systems such as *Vista*'s UAC, has long been criticized as something of a weaselly way out of security obligations, passing the buck onto users who are not generally in the best position to decide what to trust. In software like this it does at least give users exposed to malware an extra chance, with the added backing of the consensus opinions. However, for many, such decisions will invariably be based on convenience and dispatch, and in many cases the default selection will be chosen without much thought or even a glance at the popup.

The best audience for these products doubtless constitutes those who make the effort to learn what they are at risk from, how it might present itself and how to protect against such attacks, and for them *DriveSentry* represents a great addition to their security arsenal, not a catch-all but certainly a handy extra layer of defence. If the less-in-the-know users could be persuaded to make the effort to learn how to use computers and networks safely, the world would be a much less dangerous place. Outside of Utopia though, this kind of hybrid multi-pronged approach, combining IDS, whitelisting and simple malcode blacklisting with global threat monitoring and self-regulation, may well be a vision of the future.

Technical details

DriveSentry products were variously tested on:

Intel Pentium 4 1.6 GHz, 512 MB RAM, running *Microsoft Windows XP Professional SP3.*

AMD Athlon64 3800+ dual core, 1 GB RAM, running *Microsoft Windows XP Professional SP2* and *Windows Vista SP1* (32-bit).

END NOTES & NEWS

Hacker Halted Malaysia 2008 takes place 3–6 November 2008 in Selangor, Malaysia. For more information see http://www.hackerhalted.com/malaysia.

CSI 2008 takes place 15–21 November 2008 in National Harbor, MD, USA. For online registration see http://www.csiannual.com/.

The SecureDubai Conference on Emerging Threats takes place 4 December 2008 in Dubai, United Arab Emirates. Sessions will engage in the devastating effects and developments of DDoS attacks and how to avoid them, email encryption and the social engineering threat communities pose to a company. For full details see https://www.isc2.org/cgi-bin/events/information.cgi?event=81.

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

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

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

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

Black Hat Europe 2009 takes place 14–17 April 2009 in Amsterdam, the Netherlands, with training taking place 14–15 April and the briefings part of the event from 16–17 April. Online registration is now open and a call for papers has been issued. For details see http://www.blackhat.com/.

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

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

The 18th EICAR conference will be held 11–12 May 2009 in Berlin, Germany, with the theme 'Computer virology challenges of the forthcoming years: from AV evaluation to new threat management'. A call for papers has been issued, with a submission deadline of 21 December 2008 for peer-reviewed papers and 14 December 2008 for non-reviewed papers. For more information see http://eicar.org/conference/.

NISC 10 will take place 20–22 May 2009 in St Andrews, Scotland. Interest in attending can be registered at http://www.nisc.org.uk/.

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

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

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

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)1235 531889
Email: editorial@virusbtn.com Web: http://www.virusbtn.com/

vb Spam supplement

CONTENTS

S1 **NEWS & EVENTS**

S1 **FEATURE**

The problem of backscatter – part 3

NEWS & EVENTS

BEST PRACTICES FOR REGISTRARS

The Anti-Phishing Working Group (APWG) has issued a best-practices advisory for Internet registrars, setting out a series of recommendations that, if followed, should help to reduce the risk and impact of phishing.

The recommendations focus on three main areas: evidence preservation for investigative purposes, proactive fraud screening and phishing domain takedown. As registrars are in direct contact with those who register domains, they may acquire key evidence about the people who register domains for fraudulent purposes that can subsequently be used to identify and prosecute them. The document therefore encourages registrars to collect and record as much of this information as possible. The document also outlines a number of 'lightweight' processes that registrars can put in place to identify fraudulent activity before domain registration takes effect and recommends best practices that registrars can use to process takedown requests quickly. The document can be downloaded from http://www.antiphishing.org/reports/APWG_RegistrarBestPractices.pdf.

EVENTS

Inbox/Outbox 2008 takes place 25–26 November 2008 in London, UK. For details see http://www.inbox-outbox.com/.

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

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

FEATURE

THE PROBLEM OF BACKSCATTER – PART 3

Terry Zink
Microsoft, USA

In the first part of this series on backscatter (see *VB*, September 2008, p.S2), we looked at what backscatter spam is and why it is such a problem. Last month, we looked at some rudimentary techniques for stopping backscatter spam, including content analysis (see *VB*, October 2008, p.S1). We also looked at some methods we could use to stop ourselves from contributing to the problem. This month, we look at another technique used to combat this type of spam: Bounce Address Tag Validation, or BATV.

BOUNCE ADDRESS TAG VALIDATION

Last month I mentioned that, when a bounce message is received, anti-spam systems could check to see whether you sent the message in the first place. BATV offers a much more secure mechanism for determining whether or not you sent the message. I won't go into the full technical details, but I will hit on the highlights.

Imagine if you could take a look at a message and determine whether or not you sent it. You can do that to a certain extent by parsing through the Received headers and checking whether they conform to your outbound email standards. For example, do they come from your email servers? Do

Who	Specified in
Originator (author)	Content - From/Resent-From
Submitter into transfer service	Content - Sender/Resent-Sender
Return address (bounces)	Envelope - Mail-From Content - Return-Path
Sending relay	Envelope - HELO/EHLO Content - Received header
Receiving relay	Content - Received header

Table 1: Structure of an email [1].

they have certain idiosyncrasies like special headers? However, you don't have to do it that way. Table 1 shows the structure of an email.

Rather than putting just the sender in the MAIL FROM field, BATV specifies that a signature (i.e. an encrypted key) should be added to the MAIL FROM field. The outgoing mail agent adds a signature to the bounce address:

Regular	BATV
MAIL FROM mailbox@domain	MAIL FROM sig-scheme=mailbox/sig-data@domain
MAIL FROM me@example.org	MAIL FROM prvs=me/tag-val@example.org

The advantage here is that the mail server receiving the NDRs and backscatter does not need to rely on the original recipient mail server to perform any verification of the sender. It can all be done at its own end:

1. The server knows that all of its outgoing mail is signed in the MAIL FROM field.
2. It receives an inbound message and it appears to be an NDR.
3. When the RCPT TO information is extracted, it should have the key value pair. If this is decrypted and validated, the message can be accepted because it **was** sent from the mail server originally. There is not even any need to filter it further. If the key value pair does not check out, the message can be discarded because it is spoofed backscatter.

The basic idea behind BATV is that it allows you to verify whether or not NDR bounces originally came from you.

BATV IN A NUTSHELL

Figure 1 summarizes how BATV is designed to work to prevent backscatter.

Note the sequence of steps:

1. I send a message to my co-worker Ritesh and hand it off through our outbound server. However, unbeknownst to me, Ritesh has recently changed his email address.
2. My outbound server signs my SMTP MAIL FROM by adding a cryptographic tag.
3. The recipient email server, mail.i_hate_spam.com, sees that the person I am delivering to, Ritesh, does not exist.
4. The mail server accepts the message, but then bounces it back with a null sender and puts the

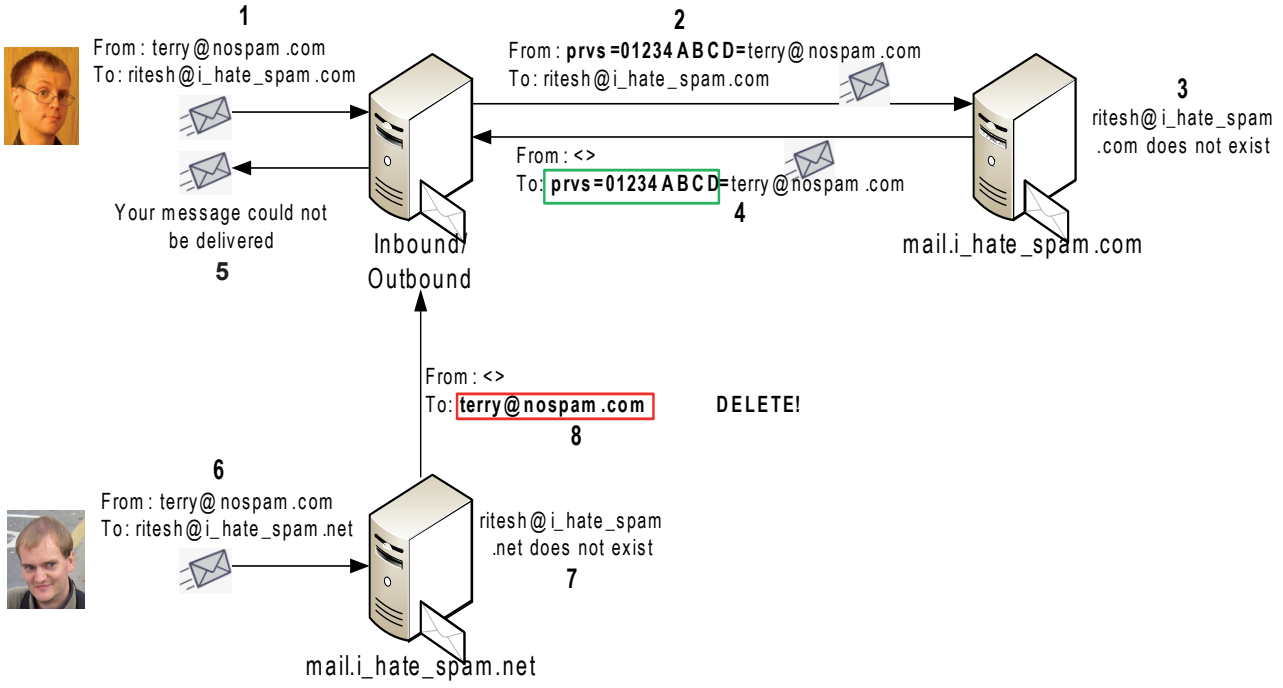


Figure 1: Summary of how BATV is designed to prevent backscatter.

original, signed, MAIL FROM information into the RCPT TO field.

5. When the message reaches my inbound mail server, it sees that nospam.com is an outbound customer. Indeed, it is my domain. My mail server determines that the message is a bounce. It decrypts the RCPT TO information which is subsequently verified, so it accepts the message and it is delivered straight to my inbox.
6. Meanwhile, evil spammer Mark Q. Spammer sends a message to Ritesh at mail.i_hate_spam.net while forging my address.
7. Mail.i_hate_spam.net accepts the message, discovers that it can't deliver it (because Ritesh doesn't exist there either) and then bounces it back to me since I appear to be the one who sent the message.
8. When the bounced message hits my inbound email server, the server sees that I am an outbound customer and that the message is an NDR. However, because the RCPT TO field is not signed, and my server knows that all genuine outbound mail from customers is signed, the message is rejected.

That's BATV in a nutshell.

BATV AND SENDER POLICY FRAMEWORK (SPF)

BATV is one of the better mechanisms available to stop backscatter. The question now is how do we use it? What potential problems are associated with BATV?

One problem is that unless you have an SPF policy that dictates a hard fail on your outbound mail, BATV doesn't necessarily work. The reason is that if you don't know where your outgoing mail is coming from, you can't necessarily say it didn't come from you if it isn't signed in a bounce.

For example, if your SPF record is this:

v=spf1 ip4:10.10.10.0/24 -all

then you know that all your outgoing mail comes only from those IP addresses. Everyone you send mail to also knows that mail from you comes only from those IP addresses and therefore your receivers should hard fail (reject) any mail that claims to come from you but which is outside of those IPs. Since you know which IPs you send mail from, you know that you always sign mail from those IPs as well. Thus, a bounce message that isn't signed means that it didn't come from those IPs; you can 'hard fail' the bounce message. It's a little like a secondary SPF check.

However, suppose your SPF record is one of the following:

v=spf1 ip4:10.10.10.0/24 ~all

or

v=spf1 ip4:10.10.10.0/24 ?all

In the former case, a soft fail '~all' means that if mail appears to come from you, but is outside your IP range, then it *probably* didn't come from you. It should be accepted, but marked as suspicious. In the latter case, a neutral fail '?all' means that if the receiver gets a mail from those IPs then it definitely came from you, but if they receive mail from outside those IPs then it may or may not have come from you – i.e. you are not entirely sure which IPs you use to send outbound mail. Thus, you neither confirm nor deny anything about mail claiming to come from you that is outside those IP ranges.

And therein lies the problem for these two cases. If you can't say for sure which IP range your mail comes from, then you can't be sure that all of your outbound mail is signed. If you can't say that all of your outbound mail is signed, then you can't reject it using BATV. An unsigned message doesn't necessarily mean that the message didn't come from you – it says so right there in your SPF policy. You'd have to parse email content in order to figure out where it came from. If you could do that, then you could implement some conditional logic because you know that messages from a certain set of IPs are signed on the outbound. This is starting to get a little convoluted, however, and it is prone to failure because you have to rely on recipient MTAs to send back all of the necessary received headers – and if you could do that (figure out where it came from by parsing and trusting the original received headers), you wouldn't need BATV.

Alternatively, you could specify that if a bounce is signed and passes a BATV check, the message should be accepted without further filtering. Conversely, if it isn't signed and you know that it is a bounce, it should be filtered more aggressively (i.e. don't de-spamify a message classification). The problem is that this takes us back to the issue of false positives; although you'll probably have fewer false positives anyhow because the ones you want are *probably* sent from your known good IP range.

I'm not really a big fan of filtering messages more aggressively, I'm just saying that you could do it this way if you had soft fail or neutral SPF policies. The Holy Grail of filtering is to accept the messages you trust and take a harder line on those that you do not. However, in my experience, being more aggressive on untrusted messages just means you add more spam points to the ones you would have caught as spam anyway, and the messages you aren't sure about just end up as false positives.

LIMITATIONS OF BATV

While BATV is a good technique, we've seen that it does have some limitations when combining it with an SPF policy. What else do we have to consider with BATV?

1. **Catch-all addresses or non-deliverable addresses**. Some MTAs will look up the recipient in the SMTP conversation. For example, in a hosted service, some companies will upload their valid email addresses and upon receiving an inbound message, the hosted service checks to see if the user to whom the sender is delivering exists. This allows the customer not to have to deal with a bounce; instead it's done upstream. Similarly, catch-all addresses will deliver non-existent mail to the catch-all instead of bouncing it.

   Because BATV changes the recipient email address on all bounces, you need to make sure that your MTA parses the BATV-signed recipient address properly. Otherwise, your MTA will receive the incoming message, check the recipient against a list of valid email addresses and say 'No, it doesn't exist because it's got this prvs=012345AbCd= in front of it, and none of my valid addresses contain that.' So, you need to make sure that you upgrade your inbound MTA to make sure it strips the leading BATV tag before performing a lookup.

The next few points are paraphrased from the Internet draft [2].

2. **Mailing lists**. BATV will cause problems with some mailing lists that identify posters by their bounce address. The list will not recognize the identical MAIL FROM addresses, because it will interpret the differing BATV attributes as part of the address. These services will either reject postings or pass them all to the moderator.
3. **Greylisters**. Greylisting is sending a 4xx-level notification to a sender which means 'Hey go away, come back later' and is based on the theory that a spammer won't return, but a legitimate sender will. A correct BATV implementation will only result in routine delays in this case. However, the result of BATV tagging MUST be a constant local-part, for a given message, and not (say) be created at delivery time such that each retry gets a different validation string, which would prevent it from ever getting through to a greylisting site.
4. **Whitelisting/safe senders**. If you send outbound mail and suddenly start signing it, people who have whitelisted your MAIL FROMs will suddenly stop recognizing your mail because the MAIL FROM will be different every day. The solution to this, of course, is to update your MTA software such that it supports BATV and is capable of stripping the BATV component of the MAIL FROM before performing sender lookups.
5. **Challenge/response systems**. Challenge/response (C/R) systems are systems where if you send an email to someone, they bounce it back to you requesting that you click a link to verify that you are a real human and not a spammer. Only once you have done that will the message be delivered to the recipient. The problem BATV poses here is that each signed message can have a different MAIL FROM so, whenever you change the keys, the C/R-protected email inbox will issue you a new challenge. This becomes very annoying for the sender.

To summarize, it is advisable to make a list of all the possible things that can go wrong before implementing BATV. Unintended consequences can cause a major headache if customers start to complain and you have to roll out a feature again.

WRAPPING IT UP

Backscatter spam is annoying. It's tough to filter because its contents can fool content filters, and it can fool end-users too.

Indeed, if your content filter could recognize an NDR and ignore the parts that typically occur in NDRs, you could filter the rest of the message normally and make the spam/not-spam classification that way.

When it comes to NDRs and Delivery Status Notifications, the key thing to remember is to treat them as a subclass of actual email. It's not marketing, it's not business mail, it's not a personal communication, it's simply a notification that mail that you sent did not get delivered in the way you expected.

We've seen a number of ways to filter the mail, some better than others. Ultimately, what it comes down to is treating bounce messages differently from regular inbound mail and making decisions based upon that special categorization of email. The rules of normal inbound filtering are modified because that's a better way to evaluate it.

REFERENCES

[1] http://mipassoc.org/batv/BATV-Intro-02dc.html.

[2] http://mipassoc.org/batv/draft-levine-smtp-batv-01.html.