MAY 2007

virus BULLETIN

Fighting malware and spam

CONTENTS

ISSN 1749-7027

IN THIS ISSUE

ANI VULNERABILITY WILL DO

The time between the announcement and exploitation of vulnerabilities continues to shrink – especially in the case of stack overflow vulnerabilities, which require very little skill to exploit. Peter Ferrie describes a prime example: the recent ANI vulnerability and its exploits.
page 4

CAT FIGHT

Although keenly aware of the descriptions and blog entries posted about his creations, the author of Nirbot seems not to be so well versed on the naming conventions used within the AV industry. Lysa Myers shares the details of this technologically unremarkable, yet reasonably successful bot.
page 9

COVERT OPERATIONS

Can a botmaster send commands covertly to a botnet of over a million zombies and control them in real time? John Aycock considers how such a botmaster's command channel might look.
page 11

vbSpam supplement

This month: anti-spam news & events, and John Graham-Cumming provides a roundup of the papers and presentations at the 2007 MIT Spam Conference.

'I believe that if the human factor is such a significant part of the problem, then it must also form part of the solution.'

David Emm, Kaspersky Lab

SECURING THE WEB 2.0

'The only constant is change' was a favourite maxim of a former employer. It's certainly true of technology, where a bewildering array of new products and versions follow one another in quick succession and the period between a product's launch and its demise seems to get shorter and shorter. Technology, of course, is only one half of the equation. The other is the human factor. On the one hand, technology enables people to do more, better, and faster. On the other hand, people drive technological change.

This applies to the Internet as well. Since its humble beginnings in 1990, the web has come to play more and more of a role in everyday life, both business and personal, and the technologies that power it have continued to evolve. Some argue that this evolution has resulted in a second generation of web services, often referred to as Web 2.0.

Although '2.0' suggests a technical release, or updated standard, there's no clear definition of Web 2.0. Tim O'Reilly, who is credited with first using the term, defines it as a 'business revolution in the computer industry caused by the move to the Internet as platform, and an attempt to understand the rules for success on that new platform.' Chief among those rules, according to O'Reilly is: 'Build applications that harness network effects to get better the more people use them.'

That said, there are technological changes too. These include AJAX (Asynchronous Java Script and XML) and other technologies that support web-based applications and the content creation and sharing characteristic of Web 2.0.

Even if Web 2.0 is dismissed as a marketing buzz-word/phrase, the changes it denotes are real enough. We've seen a transition from a web of static information stores and one-to-one relationships between sellers and buyers, to a highly interactive web in which almost anyone can post anything anywhere. In particular, there has been a massive growth in the popularity of social networking websites, such as *Bebo*, *Friends Reunited*, *MySpace*, etc.

The security implications of these changes are twofold. For one thing, the use of exploits to launch code or steal confidential data is likely to increase. We also face the prospect of malicious code that spreads from online profile to online profile, rather than seeking a home on the victim's computer.

There's also a heightened risk of identity theft. Many social networking sites have a very large number of users. Moreover, the nature of these sites means that users are predisposed to share a lot of personal information – data which is attractive to cyber criminals.

This problem is exacerbated by password issues. Users who have accounts on social networking sites are likely to be active on other sites. Many use weak passwords that reference personal information such as spouse's name, date of birth, etc. Unfortunately, few of these users have a unique password for each site.

Some people dismiss attempts to change the behaviour of employees and home users as futile. However, I believe that if the human factor is such a significant part of the problem, then it must also form part of the solution.

I'm not suggesting that there are any quick fixes. However, evidence suggests that behaviour can be reshaped over time: campaigns designed to encourage seat belt usage in cars and to discourage drink-driving have borne fruit. And this makes me wonder if we're using the wrong means to reach the people we want to educate. There is plenty of good information online, but you have to know where (and how) to find it – not easy for an inexperienced user. Maybe now it's time to shift the online security message into the offline world: a series of TV ads, like those used in anti drink-driving and anti-drug campaigns, or maybe print ads; as IT security experts, we sometimes forget that people still read newspapers. I believe that advertising safe computing practices offline could have a significant impact on the security of Web 2.0.

NEWS

A NEW GENERATION OF PANDA

Spanish security vendor *Panda Software* has announced the sale of 75% of its shareholding to southern European investment group *Investindustrial* and private equity firm *Gala Capital*.

Existing shareholders Berta Frías, Jose Maria Hernandez and Mikel Urizarbarrena retain the remaining 25% of the company and will continue to act as members of the company's management team. Although the details of the sale have not been disclosed it is believed that the deal was brokered for in the region of €100m.

The company already has a strong presence in the European small and mid-sized business market, and with its new investors on board has announced plans for international expansion and the development of new technologies in a strategy imaginatively dubbed 'Panda 2.0'.

WEBSENSE SNAPS UP SURFCONTROL

In other company news, US firm *Websense* has sealed a $400m cash deal to acquire British web-filtering company *SurfControl*. The 700-pence-per-share price paid by *Websense* – 63% more than *SurfControl*'s share price at the time of announcing it was looking for a buyer – has prompted speculation that *Websense* may not have been alone in bidding for the company. The purchase gives *Websense* a presence in the email security and hosted services markets, as well as a stronger presence in Europe and in the small and mid-sized business market, putting it in a more competitive position against large security companies such as *Symantec*, *McAfee* and *Trend Micro*.

VB100 PROCEDURE REVIEW

VB has reviewed the test procedures for the VB100 comparative testing and certification program. An updated version of the VB100 procedures document is available at http://www.virusbtn.com/vb100/about/100procedure.xml.

ERRATUM: VB100 LINUX COMPARATIVE

Upon closer analysis of the latest set of VB100 test results (see *VB*, April 2007, p.11) *VB* has regrettably discovered some errors in the detection figures published for the *Dr.Web* product. A re-run of the tests demonstrated that the product was, in fact, capable of detecting all samples in the macro, file infector, Linux, and worms & bots test sets. However, the failure to detect a small number of samples in the polymorphic test set was confirmed, as was the failure to detect three samples from the WildList test set. *VB* extends its apologies to *Doctor Web* for these errors.

Prevalence Table – March 2007

Virus	Type	Incidents	Reports
W32/Bagle	Worm	2,809,375	26.75%
W32/Mytob	Worm	2,616,941	24.91%
W32/Netsky	Worm	2,389,643	22.75%
W32/MyWife	Worm	1,031,746	9.82%
W32/Virut	File	429,045	4.08%
W32/Zafi	File	403,150	3.84%
W32/Lovgate	Worm	180,134	1.71%
W32/Mydoom	Worm	170,219	1.62%
W32/Bagz	Worm	120,551	1.15%
W32/Parite	File	68,781	0.65%
W32/Sobig	Worm	67,298	0.64%
W32/Bugbear	Worm	39,574	0.38%
W32/Jeefo	File	33,547	0.32%
W32/Funlove	File	32,966	0.31%
W32/Klez	File	28,074	0.27%
W32/Mabutu	Worm	15,134	0.14%
W32/Tenga	File	9,577	0.09%
VBS/Redlof	Script	9,364	0.09%
W32/Valla	File	7,258	0.07%
W32/Yaha	File	6,290	0.06%
W32/Womble	File	5,920	0.06%
W32/Sober	Worm	4,024	0.04%
W32/Reatle	Worm	3,226	0.03%
W32/Sasser	Worm	2,667	0.03%
W32/Maslan	File	2,565	0.02%
W32/Magistr	File	2,091	0.02%
W95/Spaces	File	1,821	0.02%
W32/Dumaru	File	1,673	0.02%
W32/Sality	File	1,600	0.02%
W32/Stration	Worm	1,529	0.01%
W32/Traxg	File	1,114	0.01%
W32/Elkern	File	896	0.01%
Others [1]		6,495	0.06%
Total			100%

[1]The Prevalence Table includes a total of 6,495 reports across 39 further viruses. Readers are reminded that a complete listing is posted at http://www.virusbtn.com/Prevalence/.

EXPLOIT ANALYSIS

ANI-HILATE THIS WEEK

Peter Ferrie
Symantec Security Response, USA

The time between the announcement of a vulnerability and the exploitation of that vulnerability continues to shrink. This is especially true when the vulnerability in question is a stack overflow, since it requires very little skill to exploit. The recent ANI vulnerability is a prime example. Before we get into that, let's find out a little more about ANI files in general.

An ANI file contains animated cursors and icons – for example, the hourglass which turns upside-down during heavy processing. Internally, ANI files are *Microsoft*'s Resource Interchange File Format ('RIFF') files. They are, in fact, little-endian versions of *Electronic Arts*' Interchange File Format ('IFF') files that were introduced to the *Amiga* in 1985. IFF files themselves were inspired by *Apple*'s OSType files that were released with the *Macintosh* computer in 1984.

IT'S 'TERIFFIC'

RIFF files contain a collection of chunks, each of which begins with a four-byte type-name, followed by the size of the chunk.

The first chunk in a RIFF file is named 'RIFF'. This is one of the two types of chunk that contain a subtype and a collection of subchunks (the other one is 'LIST').

The subchunks have the same format as chunks. The idea is that the rest of the file is a collection of subchunks within the 'RIFF' chunk. The size field is not verified (most likely because it is assumed that nothing follows the 'RIFF' chunk). Instead, *Microsoft*'s parser ensures that accesses remain within the bounds of the file by calling the GetFileSize() API and comparing file offsets against the returned value.

The 'RIFF' chunk subtype has the name 'ACON' (which may be an abbreviation of 'Animated iCON'). Very few subchunk types are supported, and some of them depend on the presence of others earlier in the file. For example, until an 'anih' type is seen, only a 'LIST' type with the 'fram' subtype and 'icon' subchunk(s) is accepted. All other types are skipped. Once an 'anih' type is seen, the 'seq', 'rate', and additional 'anih', types are also accepted.

PATCHWORK QUILT

The vulnerability that is being exploited is an unbounded copy operation to a fixed-size stack buffer. Exactly the same vulnerability in exactly the same function was found in 2004. The 'anih' subchunk contains a field that specifies the size of the data. While the data are a fixed size – 36 bytes – the value in that field is used in a copy operation. This allowed an attacker to specify an arbitrary size for the block and overflow the stack buffer. At the time, *Microsoft* patched the vulnerability by adding a requirement that the first 'anih' block is exactly 36 bytes long. It's unclear why they did it that way, because the block is copied again later, using the fixed value of 36.

For subsequent 'anih' subchunks, the data are copied to the stack buffer, then 36 bytes are copied to another buffer, and *then* there is a check that the data are exactly 36 bytes long. It is possible that the reviewer thought that this check would prevent exploitation, but by the time the check is made, the buffer has already been overflowed.

This time the patch added the same kind of check for these subsequent 'anih' subchunks before they are copied. Since the memory of subsequent 'anih' subchunks is allocated dynamically, an additional piece of code was added to free any existing block prior to allocating a new one. So we go from this:

```
if (fccType == "anih")
  copy <size> bytes to stack
  block = allocate 36 bytes
  copy 36 bytes to block
```

to this:

```
if ((fccType == "anih") && (size == 36))
  copy <size> bytes to stack
  if (block != 0) free(block)
  block = allocate 36 bytes
  copy 36 bytes to block
```

instead of the more sensible:

```
if (fccType == "anih")
  if (block == 0) block = allocate 36 bytes
  copy 36 bytes to block
```

for only a single bounded copy operation, with no need to free anything.

BREAKING WINDOWS

Despite *Microsoft*'s claims of improved security, *Vista* is just as vulnerable as every other version of *Windows*. There were supposed to be three mitigating factors, but two of them proved unsuccessful in this case.

The first is the Buffer Security Check (/GS), which also exists in *Windows XP* SP2. This is a stack protection mechanism that is designed to prevent altered return addresses from being used, by checking a value that exists

on the stack below the return address. The idea is that if the return address has been altered, then the magic value must also have been altered. However, because of the performance impact of the /GS protection, it is optional – and even then it is applied only to functions that have certain characteristics. More specifically, /GS protection is applied only to functions that contain arrays of five or more characters (ASCII or Unicode), and is not applied to functions that contain only structures with small individual fields. Since the vulnerable routine contains only structures with small individual fields, the Buffer Security Check was not applied.

Vista's second mitigating factor is Address Space Layout Randomization (ASLR), which allows an ASLR-aware process to have its contents placed at a random address in memory. The idea is to make the return address unlikely to be reached in a single attempt, and thus prevent the exploit from succeeding most of the time. However, there are two methods by which an ANI exploit can defeat ASLR on *Vista*.

The first method is not specific to ANI exploits, but applies to any stack-based exploit for which /GS does not apply. It relies on the fact that, for architectural reasons, ASLR leaves the lower 16 bits of the address unchanged. It randomizes only eight bits of the 32-bit address on the 32-bit versions of *Vista*, but even if all 15 of the available upper bits were randomized (there would be a significant performance impact to that), the vulnerability would remain. All that is required is to find an appropriate instruction within the 64kb block that holds the existing return address. Then only the lower 16 bits need to be modified for exploitation to succeed, no matter where the process exists in memory.

EXCEPTIONAL BEHAVIOUR

The second method is specific to ANI exploits, and comes into play if no appropriate instruction can be found. It relies on the fact that a Structured Exception Handler (SEH) is installed by the caller of the vulnerable function.

If an exception occurs, the SEH gains control, but this particular SEH ignores the error and returns success. The process does not crash, and the user will not notice that anything went wrong. The result is that an attacker can send multiple malicious files to the vulnerable function, each with a different return address. Since there are only 256 possible combinations, it becomes a trivial matter to brute-force the correct address and compromise a vulnerable machine.

The only mitigating factor that stands any chance of success is Data Execution Protection/No eXecute (DEP/NX), which is a method for marking a region of data, such as the stack, as non-executable. The problem is that it works only if it is enabled for the process, and by default, DEP/NX it is not enabled for 32-bit *Internet Explorer* (which is the most likely attack vector), even if hardware-backed DEP/NX is present. A simple attack will attempt to execute directly from the exploited buffer, which DEP/NX will prevent. However, it is possible (though not easy) to craft the stack to execute the VirtualProtect() API on the exploited buffer first, and then to execute the exploited buffer itself. DEP/NX cannot prevent such an action.

Internet Explorer can be exploited easily because it supports animated cursors. According to a *Determina* advisory, *Firefox* is vulnerable too, despite the fact that it does not support animated cursors. However, given the fact that icons can be animated, and that there are multiple paths to the same code (LoadCursor(), LoadIcon(), and LoadImage(), and perhaps things like CopyImage(), GetCursorFrameInfo(), and SetSystemCursor()), it seems highly likely that *Firefox* can be coerced into calling an appropriate API. *Windows Explorer* is exploited automatically without user interaction when browsing a directory that contains a malicious file, because *Explorer* parses the file in order to display the icon.

OOPS I DID IT AGAIN

Continuing the long tradition of attackers who don't seem to understand what they're doing, we saw a collection of odd attempts at exploiting this vulnerability. The funniest one contained the 'LIST' chunk name spelled backwards. This may have been the result of bad disassembling, but the other chunk names were correct, which made the misspelling very perplexing.

There were also chunks with odd-sized blocks, but this is legal and perhaps was used as a detection bypass. In any case, the only requirements for exploitation are two 'anih' blocks, of which the first must be in the correct format (36 bytes long, frame and step count less than 65,536, flags bit 0 set to specify a *Windows* cursor or icon, etc.) and the second must contain the exploit.

CONCLUSION

So what have we learned from all this? The first ANI vulnerability was the result of bad code. The second ANI vulnerability was the result of more bad code. The way to patch bad code is to remove the bad code, not to add new bad code that hides the old bad code.

A more pertinent question would be: what has *Microsoft* learned from all this?

VIRUS ANALYSIS 1

BEYOND VIRTU(E) AND EVIL

Víctor M. Álvarez, Mario Ballano
PandaLabs, Spain

File infectors represent only a small percentage of the new malware we receive in our virus lab every day, but cavity, polymorphic, entry point obscuring and memory resident infectors are even rarer. This is the case of W32/Virtu (a.k.a. W32/Virutas or W32/Virut), a virus that has been causing trouble in some corporate networks over recent months.

W32/Virtu is not really a new virus, it is just a remake of the almost ancient Tenrobot (or Netrobot) family. However, it does introduce some interesting changes and new techniques that are worth looking at.

ON THE PREJUDICES OF EMULATORS

Being a polymorphic virus, emulation should be the logical approach to detect and disinfect W32/Virtu. Indeed, this seems to be what the virus writer thought when he was creating it, so he decided to make our job a little harder by implementing some anti-emulation tricks. These are executed at the beginning of the virus execution path in some of the infected files.

Anti-virus emulators must be able to cope with *Windows* API calls in order to handle modern packers and polymorphic viruses. Most of them already emulate the behaviour of many of the functions commonly used in unpacking code, such as LoadLibrary and GetProcAddress. However, some emulators assume wrongly that API functions will always be invoked with the correct parameters. These assumptions are exploited by W32/Virtu, which performs bogus calls to arbitrary *Windows* API functions, passing deliberately incorrect parameters to them. As a result, some emulators get confused and stop emulating the virus code too early. All of this assumes that the emulator implements the API in the first place.

However, emulation is not really necessary to detect or disinfect W32/Virtu, as we will discuss later.

THE INFECTIOUS SPIRIT

W32/Virtu infects only files with EXE and SCR extensions, also excluding files whose names begin with 'PSTO', 'WC32', 'WCUN' or 'WINC'. It also checks that the file is neither a DLL nor an executable image for the *Windows* native subsystem. Files containing a section whose name begins with '_win' are also excluded, in order to avoid infection of certain Winzip Self-Extractor Archives which have a section named '_winzip_'. To avoid infecting the same file twice the virus also checks its own infection mark, which is stored in a reserved field of the executable's DOS header.

When infecting a file, the virus increases the size of the last section to fit its encrypted body. The polymorphic decryption routine could be also added to the last section, inserted into a cavity between sections if the virus finds a suitable unused space, or could be written over the original host's entry point. The last section's attributes are modified too, gaining executable and writable flags.

THE ENTRY POINT DISPOSITION

In order to get the execution flow directed to their code, some viruses change the entry point indicated in the PE header, others overwrite the original host code at the entry point, and some use entry point obscuring (EPO) techniques to make detection more difficult. W32/Virtu uses a combination of these three approaches. When it is about to infect a file, the virus decides which one will be used.

The entry point modification and overwriting approaches are not too different from what we've seen before in many other virus families. When the virus overwrites the original entry point of the host, a copy of the overwritten bytes is stored in the encrypted part of the virus body. As always, those bytes are restored in the memory image of the host before it is executed. This means that the virus body must be decrypted in order to disinfect infected files.

In the case of EPO infection, the virus starts at the host's entry point looking for CALL instructions pointing to KERNEL32.DLL. When the instruction is found it is replaced by a call to the virus decryption routine. The original bytes overwritten by the virus are stored inside its encrypted body. When the execution flow of the host reaches the modified call instruction, the virus takes control, restores the original bytes on the host, and allows the original call to be executed.

The virus author was careful to take into account the fact that API calls can be performed in two different ways. An API call can be performed with a memory indirect CALL (opcode 0xFF15) taking as argument the address on the IAT which stores the address of the API function, or it can be done through a relative CALL (opcode 0xE8) to a JMP instruction, which in turn jumps to the corresponding API. Both cases are handled correctly by the virus.

It should also be noted that, when using EPO, the virus only intercepts calls to KERNEL32.DLL. This is because it uses the address of the intercepted function as a starting point when searching for the base address of the library. To do so, it takes the address, rounds it down to a 4KB boundary, and starts decreasing the address by 256 bytes until it finds the

MZ-PE header of kernel32.dll. When the infection is not EPO, the virus relies on the fact that the program entry point is always invoked from a call which resides in KERNEL32.DLL, so it uses the return address pushed on the stack by kernel code as the starting point to search for the base address.

POLYMORPHISM AND INTERLUDE

W32/Virtu is only slightly polymorphic. Its decryption algorithm is based on XOR or SUB operations with a variable sliding key. The polymorphic engine also generates superfluous instructions and bogus loops to slow down anti-virus emulators, and is responsible for generating the anti-emulation trick mentioned in the previous section. It doesn't use FPU instructions or special purpose instruction sets. In fact, for an anti-virus product to decrypt the virus body it is not even necessary to use emulation technology – it can easily be done by employing X-ray techniques.

Furthermore, the anti-emulation trick mentioned above, which is not under the polymorphic encrypted layer, produces code patterns that can easily be detected by anti-virus engines and considered as a symptom of W32/Virtu infection.

Polymorphism is certainly not one of the (W32) virtues of this virus.

NATURAL HISTORY OF MEMORY RESIDENCE

Due to obvious architectural differences between the *Windows NT* and *Windows 9x* operating system families, the virus undertakes different strategies to achieve memory residence depending on the platform. From this point on we will use the term '*Windows NT* family' to describe all *NT*-based versions with the exception of the original founding fathers: *Windows NT 3.x* and *Windows NT 4.0*. This is because the virus makes use of the CreateToolhelp32Snapshot API, which was first introduced in *Windows 2000*.

On the *Windows NT* family the virus performs a form of multi-process residence. It starts by creating a named shared section via NtCreateSection. The section is called 'W32_Virtu'. Then it copies part of its own code to the shared section and jumps there. It also sets SeDebugPrivilege on the running process in order to access the memory context of other processes in the system. Then it iterates over the processes list, but skips the first four, which in a typical system are: System Idle Process, System Process, Windows Session Manager (SMSS.EXE) and Client Server Runtime Process (CSRSS.EXE). For the remaining processes it creates a view of the shared section in which the virus resides in order to make its code visible to the process, and then it hooks some NTDLL.DLL APIs by overwriting the very first bytes of the functions with a call to its own routines. The intercepted APIs are:

NtCreateFile

NtOpenFile

NtCreateProcess

NtCreateProcessEx

By intercepting NtCreateFile and NtOpenFile, the virus has the opportunity to infect any file opened by infected processes. The infection is performed before passing control to the genuine API function. By intercepting NtCreateProcess and NtCreateProcessEx, the virus is also able to place its hooks into newly created processes, so they become infected from the very moment of their creation. In this case the original API is invoked first, and then the virus takes control and uses the handle to the new process to install its hooks. Finally, the virus returns control to the caller.

Besides API hooking, the virus also attempts to create a thread in the context of the fifth process of the list (remember that the first four are ignored), which is usually WINLOGON.EXE. If the operation fails, it tries with the next process. If it succeeds, it stops trying, resulting in a single thread injection. This thread has two objectives: opening a backdoor on the affected machine, and disabling the Windows System File Checker (SFC) mechanism. We provide more information about these topics below.

On *Windows 9x*, the virus follows a more bizarre path to achieve memory residence. First, it calculates the address of the undocumented function VxDCall, which is exported by ordinal on KERNEL32.DLL. The virus gets the function RVA from the memory image of KERNEL32.DLL's export table, and adds the image base to obtain the function's address. Then it reserves a chunk of memory from the shared memory area, which is a zone above 0x8000000 shared by all processes on *Windows 9x* systems. This memory chunk is reserved by invoking VirtualAlloc with undocumented flags in the flAllocationType parameter.

As in the case of *Windows NT*, the virus copies a portion of its code to the shared memory area and jumps to that code to continue the execution. At this point, it makes use of the VxDCall function to invoke the PagerRegister service from the Virtual Machine Manager (VMM). This service allows a set of routines to be registered, which are invoked by the VMM whenever a page associated with the pager is paged in, paged out, or decommited. The structure for registering a pager, as documented in the *Windows 98* DDK, is the following:

```
typedef struct pd_s {
  PFUNPAGE pd_virginin;
  PFUNPAGE pd_taintedin;
  PFUNPAGE pd_cleanout;
  PFUNPAGE pd_dirtyout;
  PFUNPAGE pd_virginfree;
  PFUNPAGE pd_taintedfree;
  PFUNPAGE pd_dirty;
  ULONG pd_type;
} PD;
```

The pager registered by W32/Virtu only specifies a routine for the field pd_virginfree of the pager-descriptor structure. This routine is invoked when a page is decommited, but has not been written to since it was committed. After registering the pager with the VMM, the virus commits one page of memory, associates it with that pager, and immediately frees the page without writing anything to it, consequently causing a call to the routine pointed to by pd_virginfree. The interesting thing from the point of view of the virus, and the real motivation behind all of this, is that this routine is invoked at ring-0 privilege level. A very uncommon method for getting ring-0.

With the absolute freedom of the ring-0 privilege level, the virus queues an asynchronous procedure call by invoking the system service QueueUserApcEx. The procedure provided by the virus is executed in the context of the kernel service process, where the virus creates a new thread. This new thread is responsible for patching the VxDCall function to intercept calls to the VWIN32_Int21Dispatch (0x2A0010) system service. It also opens the backdoor mentioned when describing its behaviour on the *Windows NT* family. In fact, the code executed by this thread is the same on both platforms, with the exception of certain platform-dependent portions which may or may not be executed, depending on the operating system version returned by GetVersion.

The interception of the VWIN32_Int21Dispatch service by patching the code of VxDCall is a well-known technique employed by other viruses such as W95/Blackbat and W95/HPS (see *VB*, June 1998, p.13). Basically, the virus scans the code of the VxDCall function (in this case 30 bytes from the beginning), searching for the signature 0x2EFF1D which belongs to a memory-indirect FAR CALL instruction. The virus modifies the destination address for the call, which is stored in a writable memory area of KERNEL32.DLL, and inserts a pointer to its own code.

When the VWIN32_Int21Dispatch service is invoked via VxDCall, the virus checks whether the caller is requesting a file opening operation through a LFN_OPEN_FILE_EXTENDED function code (0x716C). If this is the case the file is infected before passing control to the operating system service. The virus implements its own synchronization mechanism to avoid re-entry due to file opening requests generated by the virus infection routine.

OH! SCHOLARS

As mentioned above, the virus disables the SFC mechanism implemented on some *Windows* versions to enable it to infect system protected files. This is carried out by the thread injected in the fifth process of the process list, which calls an undocumented function exported by SFC.DLL with ordinal number 2. For this function to work, it must be called by WINLOGON.EXE. If it is called by any other process, it simply fails. The virus author made the wrong assumption that WINLOGON.EXE would always be the fifth process of the list. This is true in many cases, but not all. It would be very easy to include the necessary code to determine which process on the list is really WINLOGON.EXE, but the virus author simply took the shortest way.

BACKDOOR VIRTUES

Besides being a file infector, W32/Virtu also behaves as an IRC bot which allows a remote attacker to execute arbitrary programs on the infected machine. The resident part of the virus tries to establish a connection to the IRC server proxim.ircgalaxy.pl and join the &virtu channel with a random nick. Once there, it waits for private messages of the form:

```
!get http://<URL here>
```

Whenever a message like this is received, the virus downloads the file from the specified URL to a temporary file and executes it. However, at the time of writing this article the IRC server was down, rendering this part of the virus useless.

AFTERSONG

O noon of life! O time to celebrate!

O summer garden!

Relentlessly happy and expectant, standing.

Watching all day and night, for friends I wait

Where are you, friends? Come! It is time! It's late!

These extracts from Friederich Nietzsche's *Beyond Good And Evil: Prelude to a Philosophy of the Future* appear inside W32/Virtu. Perhaps the virus author was simply trying to spread the German philosopher's work – in which case it's a shame that, being within a double-encrypted virus body, it will not have a very broad audience at all.

VIRUS ANALYSIS 2

NIRBOT: TARGETED ATTACKS GET PERSONAL

Lysa Myers
McAfee, USA

There has been a considerable amount of discussion in the news recently about a new bot family which refers to itself as 'IrnBot'. By all accounts, this is a relatively unremarkable bot technologically. It is less functional than your average Sdbot variant, and it utilizes fewer different exploits to get onto machines. And yet, it would appear that this tactic has been reasonably successful. New functionality has been added over time, but the bot has kept things relatively simple.

This bot gains its notoriety primarily from the cat fight in which the author has been engaged. The author seems to be keenly aware of the descriptions and blog entries posted about his creations, but not so well versed on the naming conventions used within the AV industry. The names chosen by the various AV vendors seem to have stuck in his craw, and since they have not been 'corrected', he has been picking fights. This has taken the form of comments within the bots, his choice of IRC server names, and his choice of vulnerabilities to exploit.

THE EARLY DAYS

The first variants of Nirbot were backdoor trojans rather than worms, and were simplistic even by IRC backdoor standards. When executed, they copied themselves to the Windows System directory and created a registry entry in the 'Run' key. The trojan would then contact an IRC server to join a predefined chat room. It could then be instructed to do things such as carrying out DDoS attacks, adding and deleting files, downloading files, capturing keystrokes, and uninstalling itself. The chat room and IRC server names at this point were random, or at least nothing that would arouse much suspicion.

From there, functionality was added to disable a list of process names so that detecting the trojan's presence would be marginally more difficult. Then, mirroring the history of Sdbot, scan commands and lists of usernames and passwords were added so that it could copy itself to open or weakly protected shares. At this point, the author seemed to turn his focus on vulnerabilities, adding the *Windows SQL Weak Password* vulnerability (MS00-035), *Microsoft Windows Server Service Buffer Overflow* (MS06-040), and the *Symantec Client Security and Symantec AntiVirus Elevation of Privilege* vulnerability (SYM06-010).

After increasing the bots' spreading capabilities, functionality regained the author's focus. The bots were made capable of sending spam, launching TFTP and HTTP servers and proxies, as well as stealing CD keys for popular programs such as *Windows*. He also added functionality to thwart debuggers so that getting memory dumps would be more difficult. Later variants were tied to other malware such as W32/SpotFace, which gave them the capability to spread through IM.

WHAT'S THE BIG DEAL?

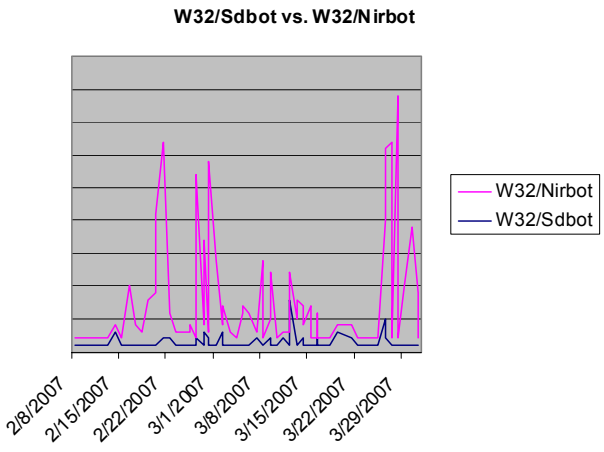
The choice of the *Symantec* AV vulnerability is what truly set this bot family apart. This gives it a potentially different set of machines to infect – people who have AV software installed, and who perhaps have a habit of patching their machines at least semi-regularly. Sdbots and their derivatives have an ever-dwindling number of infectable machines at this point, as they've been using much the same vulnerabilities for some months. Nirbot is not trying to encroach much on that territory. By focusing on machines with *Windows Server*, *SQL*, or *Norton AV*, Nirbot is going after a smaller subset of machines to begin with. Most of the machines that would be infected through these vulnerabilities will be enterprise machines which should, in theory, have better overall protection.

There is some speculation that the spike in botnets seen in March on *ShadowServer*'s yearly charts is due to Nirbot's activity [1]. It has provided something of a revival in botnet activity, since the traditional bots such as Sdbot are no longer generating the flood of activity they have in the past.

That being the case, it is clear that people are not patching all their software, regardless of whether they patch *Windows*.

A HISTORY IN COMMENTS

This is not the first time a virus author has been displeased with the AV industry's choice of names, nor is it the first



time that such a complaint gained them more media attention. In 2001, comments in later Nimda variants indicated that the author was displeased with the AV industry's choice to ignore his choice of name – Concept [2].

From the first variants, the Nirbot author has included comments apparently intended for AV researchers. Their progression shows us that this author is one who keeps close tabs on blogs and media coverage of his creations, as well as monitoring detection by the various AV vendors. He seems to share the same misconception as the Nimda author about how viruses are named. Clearly, he either hasn't been keeping tabs on any creations aside from his own, or else he's new to the game.

The first comments to appear were quite friendly in tone:

```
Dear Antivirus Employee: I see you have found one of
my creations. If you must make a definition please
call it IrnBot. Lots of Love, Author of IrnBot

ATTN ANTIVIRUS EMPLOYEE: If you're going to name my
very nicely coded modular bot, at least give it the
proper name of "IrnBot". Lots of love, Author of
IrnBot

Dear antivirus employee: well it's been an
interesting week, it's been a good battle.
P.S. The name is IrnBot, make corrections now please.
```

By the beginning of the third week in February it was clear that the author was keeping tabs on the detection of his creations, and that he was not happy with what he saw. He had quickly progressed from a very friendly tone, to pleading, to being downright vitriolic:

```
Hello antivirus employee, I must protest your virus
naming system, it isn't very accurate.

I as a malware author believe that I deserve the
right to at least have my creations named properly;
like come on, I'm the one who keeps your ass in
business. Anyways this isn't "RinBot", "VanBot" or
"NirBot"; the correct name is "IrnBot". Thank you
Panda Antivirus for getting this correct. For the
rest of you, I hope you read this and make the
correction, or ELSE.
```

As people within the security industry started trying to shut down his botnets, the author started taking the attention quite personally. He threatened DDoS attacks against the *SANS* website and started using IRC servers named to insult both *Symantec* and *SANS*. Stephen Doherty was the author of the first Nirbot/Rinbot description for *Symantec*, which seems to have earned him a special place in the malware-author's heart.

```
You better f*** off SANS.org especially that Johannes
Ullrich (jullrich@***, ***-***-****) and Kevin Hong
(khong@***.**, +**-*-**-***). I really don't have
anything against you, just p*** off alright?

Dear Antivirus Employee: It's been a rough week here
at base camp, but we will prevail. Lots of love,
author of IrnBot. P.S. F*** off Symantec.

Sorry about the hospital computers :(
;
Dear Symantec: For years I have longed for just one
thing, to make malware with just the right sting, you
detected my creation and got my domains killed, but I
will not stop, I can rebuild. P.S. F*** you a**holes,
especially Stephen Doherty who is the biggest f*****
I know of.
```

The comment about the hospital seems to pertain to a particularly disruptive Nirbot infection in the Quebec health care system in mid-February which effectively knocked out its network for three days. Some hospitals had switched to VoIP phones, which were also knocked offline [3].

On 1 March 2007, *CNN* was infected with one of the Nirbot variants [4]. Later that week, comments appeared that were a mock interview between the malware author and *CNN*. They make it clear that the author has a rather distorted view of what fraud entails, and the effect his creations have on computers and networks. He implies it's only affecting old or ignored machines, and actually doing them a service by removing other malware:

```
Tonight on CNN: An interview with the author(s) of
Rinbot. Who are you? Hacker(s). Are you actually
disgruntled? No. Then why are you actively going
after Symantec? The worm is designed for getting the
highest yield of computers infected, not to aggravate
Symantec; there is no hate. So why attack the
Symantec anti-virus program? A lot of businesses and
universities run the application, making it a prime
target for exploitation. Are you aware that your worm
is crippling computer networks? Yes that can happen
on slow networks or networks with many computers; the
worm also searches and removes other worms from the
system, acting as a small anti-virus program if you
will. If you wish not to have those problems keep your
software updated. Why did you taunt Symantec and other
security companies? They were the first to list the
worm on their site and try and get servers shut down.
What do you intent to use the infected computers for?
Nothing very malicious; no fraud or anything like that.
What is the real name of the worm and how did you
come up with it? The real name is IrnBot, it is named
after a popular soft drink called IrnBru. Thank you
for your time author of Rinbot. You are very welcome
CNN, thank you for the opportunity to explain.
```

The soft drink he mentions, *Irn-Bru*, is a soda that touts itself as 'Scotland's other national drink' [5]. It has been around in one form or another since the early 1890s, and until recently outsold *Coca-Cola* in Scotland.

A few days after the *CNN* incident, *Offensive Computing* posted a neutral discussion of this variant, including the comment above. In response to this, the Nirbot author started giving 'shout outs' to them by referencing them in channel names [6].

In what apparently was his last comment, the Nirbot author waxes extra-bilious about a blog posted by Pedro Bueno, discussing the addition of functionality to try to thwart reverse engineering [7]. The author's choice of IRC server

names switches from insulting *Symantec* and *SANS* to insulting Bueno directly.

```
Looks like Pedro Bueno is getting smart at least
that's what he thinks. Apparently he added some new
entry to his super-hyper-extra-f*****y lame blog.
Which, by the way, does not prevent in ANY way the
infection of your computer. Pedro Bueno is gay, gay,
gay! (For Richard Simmons, that is.

P.S. If you were actually anything other then a
complete d***** bag you would realize that it was not
intended to stop antivirus researchers, just your
everyday script kiddy. Yours truely, Author of
IrnBot.
```

At the time of writing, since the media hype and blogging on the Nirbot family has died down, the author seems to have ceased commenting his creations. Variants continue to be discovered by the handful each day, and show no signs of abating. But it appears, for now, that the cat fight has ended.

CONCLUSION

This virus is really nothing new in terms of general tactics, though it's notable how successful it has become with less functionality than the usual crop of Sdbots. It would seem, in terms of both spreading and notoriety, that this virus author has done quite well in targeting AV vendors.

While a policy of not using virus writers' intended names is laudable, it does little to curb media interest in this sort of story. It *is* likely that this policy does discourage some script kiddies from trying to gain notoriety, even when it occasionally backfires as in this example.

In terms of the *Symantec* vulnerability used by this threat, there is not much that can be done aside from continuing to use secure programming practices. It is quite clear from the proliferation of bots using years-old vulnerabilities, that some users are reluctant to apply patches even when they're offered automatically.

REFERENCES

[1] http://www.shadowserver.org/wiki/pmwiki.php?n=Stats.BotCounts.

[2] http://archives.neohapsis.com/archives/isn/2001-q3/0400.html.

[3] http://www.unixadmintalk.com/f17/ot-quebec-health-care-virus-264596/.

[4] http://money.cnn.com/2007/03/01/news/companies/virus/index.htm.

[5] http://www.irn-bru.co.uk/home.html.

[6] http://offensivecomputing.net/?q=node/375.

[7] http://www.avertlabs.com/research/blog/?p=216.

FEATURE

COVERT ZOMBIE OPS

John Aycock
University of Calgary, Canada

Time for a thought experiment. An evil botmaster wants to take over the world, as evil botmasters are wont to do, and a botnet of a million zombie computers has been amassed for this purpose. Can the botmaster send commands covertly to the zombies and control them in real time?

This is not necessarily a hypothetical question. A large-scale DDoS attack could be re-aimed dynamically at different targets, or a physical attack could be accompanied by an Internet attack that changes dynamically to cause confusion. It makes sense to consider in advance how the botmaster's command channel might look, so that it could be detected and disrupted if necessary.

Unless the botnet is meant to be obvious and short-lived, there is a set of severe design constraints on communication for the botmaster. Communication must be covert; it should reach a large majority of infected machines; it must be scalable; transmissions should be limited; it should be very hard to trace the communication source; it should be resistant to the insertion of false signals; it should be sustainable over a long period of time; it should be real-time (or close to it).

Some aspects can be handled easily with existing techniques. For example, resisting false signals can be achieved by applying public key cryptography to digitally signed commands [1]. If the botmaster encrypts commands with a private key, then the corresponding public key – distributed with the malware that created the zombies – can be used both to decrypt the command and to verify that the command came from the botmaster. Longer commands, following the usual wisdom for digital signatures, would encrypt/decrypt a digest of the botmaster's command for performance reasons.

Other communication aspects require more analysis. There would seem to be a direct relationship between the traceability of the botmaster and the degree of responsiveness/interactivity the botmaster experiences when controlling the botnet. Consider two extreme points:

1. The botmaster prerecords commands, and places them in some well-known location. The zombies could periodically poll for new commands. There is no interactivity for the botmaster, and low responsiveness due to the time lag between command recording and realization. However, the botmaster is very difficult to trace, and need not even be connected to the Internet when an attack occurs.
2. The botmaster broadcasts commands continuously to the botnet. Assuming sufficient bandwidth, this would yield the highest interactivity, with responsiveness

limited only by network latency. The disadvantage is that a single source pumping out continuous network traffic would be relatively easy to trace. There is legitimate work related to this in the area of music performance, where systems have been constructed to enable musicians to perform together across a network [2–5]. Not surprisingly, overcoming network latency is the major technical hurdle. The closest system to what a botmaster would need is the 'conductor architecture' [3], where one conductor sends a global signal to multiple musicians upon which they can synchronize. The similarity ends here, though: network music performance is not intended to be covert, nor designed to scale beyond a small, finite number of musicians.

Between these points lie many feasible methods of communication, depending on how much loss of responsiveness and interactivity the botmaster can tolerate.

There are two key goals for the botmaster to accomplish. First, the amount of communication from the botmaster must be reduced. This helps increase scalability and reduce traceability. By connecting the zombies into multiple small botnets, the botmaster needs to send commands to only a limited number of botnet command-and-control machines, instead of every single infected machine; commands propagate from botnet to botnet. In effect, the botmaster would have a network of botnets, a 'super-botnet', which can be constructed automatically to resist countermeasures, yet remain highly receptive to commands [6].

Second, the propagation of commands to individual infected machines must be hidden. This helps the zombies avoid detection for longer periods. HTTP is an excellent candidate through which infected machines can poll their botnet's command-and-control server (which would run an HTTP server on port 80) for new commands. This happens already [7], but it can be done much more covertly.

Using HTTP for intra-botnet communication has definite advantages: natural cover traffic generated by real users, carte blanche to pass through egress firewalls, and automatic leveraging of web caches. However, polling an HTTP server for commands at frequent, regular intervals is not typical user behaviour. What *is* typical behaviour has been studied extensively [8].

If zombies were designed to be covert and exhibit HTTP traffic characteristics typical of users, there would be implications for the botmaster sending commands, which include:

- Time. HTTP traffic has been shown by many studies to peak during the daytime [9–12], and zombies would need to shape their traffic accordingly. Interestingly, a diurnal pattern has also been noted in botnet communication [13], but it was rationalized by saying that 'many users turn their computers off at night'. With an increasing number of always-on computers, the possibility cannot be ignored that surreptitious malware may mimic the diurnal cycle deliberately to delay detection. The implications for a botmaster are that interactivity and responsiveness cannot be expected to be consistent across the globe; zombies exhibiting diurnal behaviour would respond faster during the day, and the physical location of zombies would become a factor.
- Transfer size. HTTP traffic can be broken down in various ways, but one potential concern for a botmaster is how large a command can be transferred to a zombie without raising suspicion, i.e. not looking like normal HTTP traffic. There have been many studies that gather statistics about the median file size transfer, the median size of HTML files, and the median HTTP transfer size [12,14–18]. In this data, it is *highly* unusual to see a reported median under 2K; most report 2K or higher. From this we conclude that a median size of 2K commands can be used reliably by botmasters, which is more than sufficient to contain a short command. Note that this doesn't preclude larger transfers, like executables, but just means that they may have to be broken into multiple pieces.
- Transfer frequency. Retrieving the complete contents of one web page for rendering may result in a burst of discrete HTTP transfers, such as the download of an HTML file followed by the fetching of inline images. We assume that such rapid-fire retrieval is not indicative of how often an infected machine can poll for new commands covertly; instead, the time that would normally elapse in between complete web page retrievals is of interest over the long term.

  This time has been measured in various ways. Reported medians are 11s [17] and 15s [16], with heavy-tailed distributions that result in larger mean times of 47s [17] and 81s [19]. A definite conclusion about covert zombie behaviour is harder to draw from this data. An average polling rate of one minute appears likely, so long as plenty of time variations are introduced artificially. However, infected machines will not be polling in lock-step with one another. This means that the average polling rate should be taken as a worst-case indication of overall response time to the botmaster's commands.
- Domain names. Phishing URLs and *Google*'s cache notwithstanding, it is fair to say that a large majority of URLs specify the HTTP server using a domain name, which must be mapped into an IP address; typically this mapping is done via DNS queries. Although not part of the HTTP protocol *per se*, DNS lookups are thus a

characteristic of normal HTTP traffic, and covert zombie communication would have to exhibit this too.

Defensively, it is tempting to try and block or corrupt the zombie's HTTP traffic, but a false positive when detecting zombie communication would affect user HTTP, a high-visibility error. And, even if good and bad HTTP could be separated, it would have to be separated on a very large scale to be effective, because there would be numerous distributed HTTP servers rather than a small number of centralized ones.

Are DNS lookups the Achilles' heel of covert zombies? Security work has been done correlating DNS lookups with subsequent connections [20]; this could be applied to flag computers with consistently anomalous DNS behaviour. Zombies that get DNS lookups correct – and could avoid anomaly detection – would be relying on the DNS infrastructure. But this may not afford effective detection either; DNS caches, for example, may prevent suspicious queries from reaching a detection system.

While not giving complete responsiveness and interactivity, it seems to be within the technical reach of botmasters to have some degree of dynamic, covert control of large numbers of zombies. Users are yet again a key element, not as an infection vector, not as victims, but as the model of behaviour to which covert zombies must conform.

REFERENCES AND NOTES

My research is funded in part by a grant from the Natural Sciences and Engineering Research Council of Canada. Helpful suggestions were made by Andrew Warfield, Carey Williamson, and a reviewer.

[1] Schneier, B. (1996). *Applied Cryptography* (2nd ed.). Wiley.

[2] Chafe, C., Wilson, S., Leistikow, R., Chisholm, D., & Scavone, G. (2000). A simplified approach to high quality music and sound over IP. In *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFX-00)*.

[3] Bouillot, N. (2004). The auditory consistency in distributed music performance: a conductor based synchronization. *Information Science for Decision Making*, *13*.

[4] Gu, X., Dick, M., Kurtisi, Z., Noyer, U., & Wolf, L. (2005, June). Network-centric music performance: practice and experiments. *IEEE Communications Magazine*.

[5] Kurtisi, Z., Gu, X., & Wolf, L. (2006). Enabling network-centric music performance in wide-area networks. *Communications of the ACM*, *49*(11).

[6] Vogt, R., Aycock, J., & Jacobson, M., Jr. (2007). Army of Botnets. In *Network and Distributed System Security Symposium 2007*.

[7] Ianelli, N., & Hackworth, A. (2005). Botnets as a vehicle for online crime. CERT Coordination Center.

[8] Pitkow, J. E. (1999). Summary of WWW characterizations. *World Wide Web*, *2*(1–2).

[9] Crovella, M. E., & Bestavros, A. (1996). Self-similarity in World Wide Web traffic: Evidence and possible causes. In *Proceedings of the 1996 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*.

[10] Gribble, S. D., & Brewer, E. A. (1997). System design issues for Internet middleware services: deductions from a large client trace. In *Proceedings of the USENIX Symposium on Internet Technologies and Systems*.

[11] Abdulla, G. (1998). *Analysis and modeling of World Wide Web traffic*. Ph.D. thesis, Virginia Polytechnic Institute.

[12] Williams, A., Arlitt, M., Williamson, C., & Barker, K. (2005). Web workload characterization: ten years later. In X. Tang, J. Xu, & S. T. Chanson (Eds.), *Web content delivery*. Springer.

[13] Dagon, D., Zou, C., & Lee, W. (2006). Modeling botnet propagation using time zones. In *13th Annual Network & Distributed Security Symposium*.

[14] Bray, T. (1996). Measuring the Web. *Computer Networks and ISDN Systems*, *28*(7–11).

[15] Woodruff, A., Aoki, P. M., Brewer, E., Gauthier, P., & Rowe, L. A. (1996). An investigation of documents from the World Wide Web. *Computer Networks and ISDN Systems*, *28*(7–11).

[16] Mah, B. A. (1997). An empirical model of HTTP network traffic. In *Proceedings of INFOCOM '97*, *2*.

[17] Abrahamsson, H. (1999). *Traffic measurement and analysis*. Technical report T99:05, Swedish Institute of Computer Science.

[18] Barford, P., Bestavros, A., Bradley, A., & Crovella, M. (1999). Changes in Web client access patterns: characteristics and caching implications. *World Wide Web*, *2*(1–2).

[19] Vicari, N. (1997). *Measurement and modeling of WWW-sessions*. Technical report 184, Institute of Computer Science, University of Würzburg.

[20] Whyte, D., Kranakis, E., & van Oorschot, P. C. (2005). DNS-based detection of scanning worms in an enterprise network. In *12th Annual Network & Distributed Security Symposium*.

PRODUCT REVIEW

TREND MICRO PC-CILLIN INTERNET SECURITY 2007

John Hawes

Founded in California in 1988, *Trend Micro* is by far the largest security company based outside the US, with its headquarters in Tokyo and its founders hailing from Taiwan. In its home market in Asia *Trend* holds a comfortable market share, and its global profile continues to expand at the expense of its better-known rivals – its brand recognition having been aided by the company's successful bid for the *Microsoft Hotmail* scanning deal in late 2004.

Alongside *PC-cillin*, which is the company's flagship home-user product, *Trend* produces a broad range of security solutions including corporate desktop and server products, gateway software and appliances. Among many innovations supported by the company is the popular *HijackThis* spyware-spotting system, and the *Housecall* online scanner has also proved a useful and successful offering. The company maintains a global network of research labs, with several in Asia and Europe and one in the US.

The *PC-cillin* brand has been around for some time and continues to be applied to the company's *Internet Security* suite, while in Japan the same suite is known as *Trend Virus Buster Flex 2007*.

WEB PRESENCE, INFORMATION AND SUPPORT

Trend Micro is the proud owner of the www.antivirus.com domain name, which redirects to the company's main US site. On visiting the more predictably named www.trendmicro.com the user is redirected to a locale-specific site – although I noted that the remarkably thorough list of countries provided in the 'Worldwide' selection does not quite follow through on the (implied) promise of pages tailored to the country/language in question.

The main US homepage is a little slicker than its UK and European counterparts, which seem somewhat pared down and lacking in substance in comparison, and also more closely resembles the Japanese site.

At the bottom of the homepage is a block of floating links to popular keywords. This includes a nod to a few more technical issues, although the site is mostly dominated by more sales and marketing-focused content. The standard datasheets on individual products are provided, as well as advice on selection of the appropriate solution for a given setup. This goes as far as providing a rather fun 'Solution Recommender', which gathers details on your environment and presents the dangers and ways of minimising them with plenty of cute graphics and simple terms.

Finding support content is a lot easier here than is often the case on vendor sites, with prominent support links provided in each product section. These lead primarily to a good selection of knowledgebase and FAQ-type solutions to the more common issues associated with the software, with further contact details (phone or email) available for those with more difficult problems.

Delving further into the technical content of the site reveals a comprehensive malware encyclopaedia. This is somewhat tricky to find on some versions of the site, but it has proved regularly to be a useful resource in my own research and features a wealth of detail on many issues. The encyclopaedia often includes in-depth malware removal instructions and in many cases funky graphics and charts to illustrate attack vectors and infection processes for the more complex or common threats. There is also a lab blog, which puts an even more light-hearted spin than usual on the latest issues and discoveries.

Services are becoming a big part of *Trend*'s online offering, with many of them integrated into the product as part of the *TrendSecure* system (discussed later on). One function that is freely available to all is the *Housecall* online scanner. This is an excellent implementation of an online scanning solution, and it suffers from fewer of the browser incompatibilities and less general bugginess than many of its rival services. *HijackThis*, a registry and settings analysis tool, is also made available free of charge, having recently been acquired by *Trend* from its original developer who was unable to continue maintaining it without funding.

INSTALLATION AND SETUP

Installing *PC-cillin* is a pretty standard operation, although while testing on some older systems I found I had to

upgrade my browser to get through the process (the product requires *Internet Explorer 5.5* or later, and supports platforms released since *Windows 2000 SP4*, including 32-bit and 64-bit versions of *Vista*).

Before any software is installed locally a brief scan is run to ensure the machine is safe, an activation code is required, and options are presented allowing a non-default installation location, and the choice of a normal or minimal installation (minimal in this case apparently being everything except the firewall component).

Permission is also requested to transmit data for inclusion in software and website filtering databases before the installation proper begins. Once complete, a reboot is required and for once the checkbox confirming that rebooting is OK is not checked by default, which is doubtless a bonus for the less attentive user.

At this point some useful information is thoughtfully provided, which is that the main product interface can be launched by double-clicking the blue-and-turquoise pill in the system tray. This is a concept which some other vendors, whose products seem to revel in stealth when it comes to accessing their controls, would do well to consider.

After the reboot, my test lab machines popped up blue warning boxes to inform me that the lack of a web connection left the product unable to update, thus rendering protection less than optimal. A link in the popups led, not as I had expected to the network properties page, or to some stark warnings to fix the problem, or even (as I have seen a few times in the past) to a blank webpage confirming the lack of connection. Instead, some simple, clear advice was provided on investigating the problem, along with some simple solutions.

Connected machines at this point downloaded and applied updates, which were fairly large at around 30 MB, but

which came down and installed themselves pretty quickly. With this vital step either completed or avoided, I was ready to look at the product itself.

OPERATION AND DOCUMENTATION

When the pill icon is first clicked on, a tutorial is offered, giving a brief overview of the product. Beyond a list of newly added features for this latest version, the overview was of little interest though, simply running through the available buttons and tabs and explaining what they are used for – most of which can easily be divined from the interface itself.

The interface is clearly and simply laid out, in a style reminiscent of several other such combined suites, which is perhaps an indication that this format is becoming something of a standard – clearly a good sign for users who value consistency in their user experience. There are a few oddities due to the fact that the product design includes an element of the trendy: the real window borders are hidden, which results in an invisible strip along the top blocking some dragging movements (and affecting screenshots).

Buttons are large and colourful, and much of the product features chunky icons which occasionally dwarf the functions they illustrate. On one version of the website a ticking-cog progress display took up much of the browser window while the site processed a request, but the spinning discs used to show progress in the product itself are less intrusive.

The main interface presents a series of tabs down the left-hand side for the major modules, malware controls, network controls, web and email settings, and 'other settings' (updates, subscriptions, access to the startup tutorial and logging).

Each section is further subdivided to provide information about and configuration of various aspects of protection, generally with simple on/off controls immediately to hand and more detailed settings windows available for fine-tuning, setting up of block/allow lists and so on.

The layout seems intuitive, although experience with similar products may have biased my view on this. Few of the functions seemed hard to find (with the exception of one item, described in the next section, that is notable by its absence), and few of the controls on offer were hard to fathom. The interface is also fairly responsive, with the occasional pause, accompanied by the spinning circle, as tabs are repopulated with fresh information, but these lags never intrude too much on one's time.

Should users find themselves at a loss, the 'Help & Support' tab is clearly marked on the left, and filled with options to

connect to the web for knowledgebase articles or virus descriptions, both discussed earlier.

The local help is copious, clear and pleasing to the eye, with its numerous diagrams and splashes of colour to break up the text. However, it suffers from a lack of context-based access in the main areas of the interface – for help with a particular topic, one must return to the help tab to open the documentation (unless you have figured out how to restore the window border).

Direct links to help on a given section are, however, provided on the more complex settings and configuration dialogs. Tips within the help are rarely, if ever, linked directly to the appropriate tab or subtab required, resulting in lengthy sets of instructions on many pages as a task is walked through from the 'Open the GUI' beginning. Nevertheless, the system is nicely laid out and based on tasks rather than buttons, making information on a specific topic simple to find.

MALWARE SCANNING AND SYSTEM PROTECTION

When I came to run some scans over various test sets, I was thrown for a moment by the option for selecting an area to scan. At first this appeared only to offer full system scans, however, once I had figured it out I was able to configure it quite simply and run scans across individual folders.

The facility to scan a single file is missing from this part of the interface, and no integration into context menu is provided either, leaving users who find themselves pondering the contents of a newly downloaded or discovered file with a rather more complex process to undergo than strictly necessary. Of course, scheduled scans are also available and can be configured to cover individual files as well as drives and folders.

Scanning results were consistent with recent performances in VB100 comparative reviews, with coverage of the older collections reasonably thorough but a little behind the top performers. On more recent samples there were no such shortcomings, and coverage was excellent. Speeds were good over clean sets, and a little slow over large batches of infected files as detections are alerted on and logged, but this is unlikely to affect the average user unless they are unfortunate enough to acquire infections numbering in the thousands. *Trend* products have been conspicuous by their absence from the most recent VB100 tests, so gathering speed figures over the new speed sets would offer little value for comparison, and we must hope that an entry in the upcoming *Windows XP* test will allow scanning rates and overheads to be measured against competitive products.

Options for removal, disinfection or simple logging are generally to hand wherever a scan is set up, along with other aspects of scanning behaviour, such as the filetypes scanned or archive recursion levels (which in some products are tucked away in an obscure corner). Shutting down on-access services is achieved easily with a simple drop-down box on the appropriate tab, and invariably offers the option to restart after a given period of time.

Included on the malware protection tab are controls for the 'Venus Spy Trap', which alerts on suspicious changes to the system or software components being installed by untrusted programs. The system can be configured to monitor various areas and types of behaviour, and a whitelist of allowed programs can be set up simply by browsing to the required executable. This proved sufficient to alert on several pieces of malware not covered by definitions in an updated product.

Alongside this tool is a very interesting feature, the 'suspicious changes' listing, which offers information on any changes made to the system prior to the installation of the product, from a base level of known-safe or factory default settings. These are divided into areas such as startup items, browser settings, services, installed ActiveX controls and other software and so on. Entries within each section are graded by colour-coded risk level, from the fairly benign to the very serious security risk, and accompanied by copious detail about the change and its associated dangers. Risks deemed undesirable by the user can easily be nullified by the checking of a box and a click of the 'Undo change' button – altogether an ingenious, very helpful and wonderfully simple-to-use little item. My only quibble would be the focus on *Explorer* with no mention of other browsers, which could also present vectors for attack if not configured properly.

The final component in this section is an outbreak warning system, which presumably polls the *Trend* servers for alerts

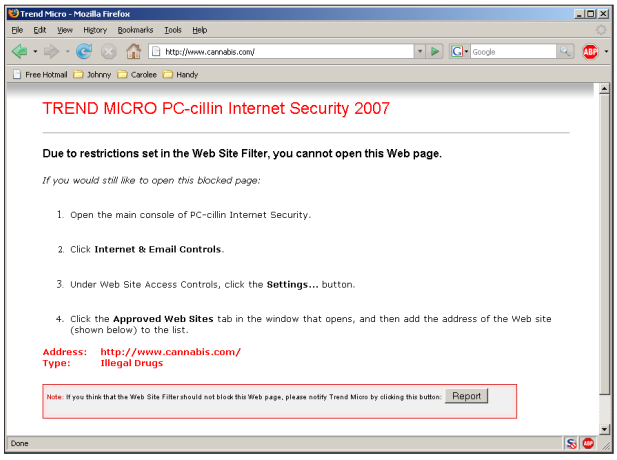
on major malware events, and ensures that updated protection is downloaded as well as warning the user of such events. Fortunately no major new attacks occurred during the testing period so the impact of these warnings could not be examined.

OTHER FUNCTIONALITY

Of course, as an Internet security suite, there is much more on offer here besides anti-virus and anti-spyware protection. A firewall is perhaps the most basic addition – an increasingly necessary defence tool. As most sensible people upgrading from a plain anti-virus product are likely to have a firewall already in place and tuned to their liking, the option to miss out this component during installation is a useful one. However, the idea of products like this is to keep as much of one's security controls together in one place and control them with an integrated interface, and here everything is fairly clearly and simply presented.

The settings tab on the firewall section opens a dialog listing types of network connection, and showing the basics of the protection profile provided for each. This can then lead to further dialogs which start with simple sliders to illustrate paranoia levels and provide further configuration fine-tuning on other tabs, covering programs allowed to connect outwards and ports open or blocked for inbound traffic, as well as more general networking settings. Explanations of what each section does are put very clearly for the untrained user, and the default settings seem generally pretty solid with no obvious attack vectors left open. Firewall modes can also be changed from the system tray menu, which also offers an emergency network shutdown option.

Alongside the firewall on the 'Personal Network' tab is a tool designed to look out for 'uninvited guests' intruding on a wireless connection, which can probe on demand or on a



schedule to see if the neighbours are stealing your bandwidth. Also in this area is a section for protecting other machines, allowing basic management of *Trend* installations on other systems in the local network, thus extending the functionality usually provided for corporate network users to the home office and multi-computer homes in general. Details of installed versions can be checked remotely, and if necessary updates can be applied before scans are run to check the remote machine's security.

The final of the three main security sections concerns the web and email, and opens with an anti-phishing section which seems to be little more than a summary of the rest of the module along with other significant items such as spyware blocking. Spam filtering is provided, but controls are fairly basic, with a simple slider for paranoia settings and no deeper configuration other than lists of approved and blocked senders.

Web filtering has slightly more complex controls, with more allow and block lists which can be set up manually and can be set to allow access to trusted sites only or merely to block the untrusted list. Known phishing sites are alerted on and a further filter allows blocking of sites by content, with a large range of categories from spyware and phishing through sex, drugs and violence down to shopping, gaming and dating. Some playing around with these settings suggested that it worked quite sensibly and thoroughly, although a few gambling sites seemed to slip through the net. A pretty thorough range of browsers and email clients are covered, including the Japanese *Becky! System* and several major webmail providers.

Finally under this heading is a data-theft protection system, which can be used to prevent items of information from passing outwards from one's computer. There is a large table of protected items, including credit card numbers, names and email addresses which can be stored for comparison against outgoing data, which is a good idea as

long as one trusts that this area is a secure basket in which to store all these eggs. This section ties in with several other online security measures provided by the 'TrendSecure' online services offering, which also includes a 'transaction guard' tool, a Java applet which provides a 'secret keyboard' for entering sensitive data from unprotected systems such as Internet cafes, as well as spyware checking in such locations, but requires administrative access to set up.

CONCLUSIONS

Overall this is an excellent product, with the basics covered well and an impressive range of innovative additions, the section for detecting potentially risky software and system changes being of particular note.

Trend has a decent record in VB100 testing, with only one failed test (due to a single in-the-wild miss in December 2006) since 2002, although the product's frequency of entry is not as good as some. Detection of *VB*'s zoo sets is generally in the mid-90% range and seems to improve all the time, demonstrating a commitment to adding quality in depth as much as to developing new ideas.

The design of the product is the main thing here though. The majority of home users need a product that can provide them with broad security coverage without excessively taxing technical involvement, and this suite certainly fits the bill with its very clear design and simple layout, and the more complex items explained in well-worded layman's terms.

More in-depth configuration is generally available for those with a little more knowledge who want to fine-tune things, although the spam filtering settings seemed less flexible than might be expected.

The colourful displays and large chunky icons may be a little much for some, but even these are, in parts, configurable. The only real flaw is the absence of that handy right-click scan, which I'm sure *Trend* will be adding in at some time in the near future.

Technical details

PC-cillin Internet Security 2007 was tested on:

AMD K6, 400Mhz, with 512MB RAM and dual 10GB hard disks, running *Microsoft Windows 2000 Professional Service Pack 4*.

Intel Pentium 4, 1.6Ghz, 512MB RAM, dual 20GB hard drives, 10/100 LAN connection, running *Windows XP Professional Service Pack 2*.

AMD Athlon64, 3800+ dual core, 1GB RAM, 40GB and 200GB hard drives, 10/100 LAN connection, running *Windows XP Professional Service Pack 2* (32-bit).

END NOTES & NEWS

DallasCon VI will take place 7–12 May 2007 in Dallas, TX, USA. For details see http://www.dallascon.com/.

The 22nd IFIP TC-11 International Information Security Conference takes place 14–16 May 2007 in Sandton, South Africa. For more details see http://www.sbs.co.za/ifipsec2007/.

The 4th Information Security Expo takes place 16–18 May 2007 in Tokyo, Japan. For more details see http://www.ist-expo.jp/en/.

The 8th National Information Security Conference (NISC 8) will be held 16–18 May 2007 at the Fairmont St Andrews, Scotland. For more information see http://www.nisc.org.uk/.

The 2007 IEEE Symposium on Security and Privacy takes place 20–23 May 2007 in Oakland, California, USA. For full details see http://www.ieee-security.org/TC/SP-Index.html.

AusCERT Asia Pacific Information Technology Security Conference takes place 20–25 May 2007 in Gold Coast, Australia. The conference focuses on IT security for CFOs, CIOs, CTOs and technical staff from government agencies, universities and industry. See http://conference.auscert.org.au/conf2007/.

The Gartner IT Security Summit 2007 will be held 4–6 June 2007 in Washington, D.C., USA. For more information see http://www.gartner.com/2_events/conferences/sec13.jsp.

The CISO Executive Summit & Roundtable takes place 6–8 June 2007 in Nice, France. The event will focus on how today's CISO can drive and integrate security into the core of the business. For details see http://www.mistieurope.com/.

CSI NetSec '07 will be held June 11–13, 2007 in Scottsdale, AZ, USA. Topics include: botnet subversion; Vista; pen testing; insider threats; forensic analysis; web-based apps; NAC; social engineering; and wireless hacking. For details see http://www.csinetsec.com/.

The 19th FIRST Global Computer Security Network conference takes place 17–22 June 2007 in Seville, Spain. For full details see http://www.first.org/conference/2007/.

IT Underground Dublin will be held 20–22 June 2007 in Dublin, Ireland. IT Underground will cover a wide range of security topics ranging from hacking techniques to OS hardening, reverse engineering, forensics and legal aspects of computer security. For details see http://www.itunderground.org/.

The Information Security Asia 2007 Conference & Exhibition takes place on 10 and 11 July 2007 in Bangkok, Thailand. For details see http://www.informationsecurityasia.com/.

The International Conference on Human Aspects of Information Security & Assurance will be held 10–12 July 2007 in Plymouth, UK. The conference will focus on information security issues that relate to people. For more details see http://www.haisa.org/.

The 2nd conference on Advances in Computer Security and Forensics (ACSF) will take place 12–13 July 2007 in Liverpool, UK. For details see http://www.cms.livjm.ac.uk/acsf2/.

Black Hat USA 2007 Briefings & Training takes place 28 July to 2 August 2007 in Las Vegas, NV, USA. All paying delegates also receive free admission to the DEFCON 15 conference, which takes place 3–5 August, also in Las Vegas. See http://www.blackhat.com/.

The 16th USENIX Security Symposium takes place 6–10 August 2007 in Boston, MA, USA. For full details and online registration see http://www.usenix.org/events/sec07/.

HITBSecConf2007 - Malaysia will be held 3–6 September 2007 in Kuala Lumpur, Malaysia. See http://conference.hackinthebox.org/.

The 17th International VB Conference, VB2007, takes place 19–21 September 2007 in Vienna, Austria. Subjects include: rootkits, automated analysis, malware in the gaming world, malware on mobile devices, anti-malware testing, spam & phishing trends and techniques, spyware, forensics, legal issues and much more. For the full programme and online registration see http://www.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

NEWS & EVENTS

EMAIL THIS!

A stark reminder of the ingenuity of spammers was received by *VB*'s web team last month thanks to a brief incident involving the *VB* web server. Suspicions of nefarious activity were first aroused when a significantly large number of emails began to overwhelm the email server. On inspection, the emails appeared to have been sent by the web server. Quick to respond, *VB*'s web developer immediately blocked port 25 of the web server to prevent any further emails being sent while he investigated the anomaly.

It transpired that a badly written perl script was the root of the problem. The code – written a number of years ago by a long-since departed member of the web team – was intended to allow visitors to the website to email articles they found particularly interesting/relevant to friends or colleagues. Unfortunately, however, the 'email this article to a friend' feature might better have been described as 'email this article to several thousand friends', since no limit had been placed on the number of addresses to which to send the message. Needless to say, the script was swiftly removed and normal service resumed.

EVENTS

Inbox 2007 will be held 31 May to 1 June 2007 in San Jose, CA, USA. For more details see http://www.inboxevent.com/.

The 10th general meeting of the Messaging Anti-Abuse Working Group (MAAWG) will take place 5–7 June in Dublin, Ireland. See http://www.maawg.org/.

CEAS 2007, the 4th Conference on Email and Anti-Spam, takes place 2–3 August 2007 in Mountain View, CA, USA. For details see http://www.ceas.cc/.

The Text Retrieval Conference (TREC) 2007 will be held 6–9 November 2007 at NIST in Gaithersburg, MD, USA. See http://plg.uwaterloo.ca/~gvcormac/spam.

CONFERENCE REPORT

MIT SPAM CONFERENCE 2007

John Graham-Cumming
Independent author, France

On 30 March, the fifth spam conference took place at Massachussets Institute of Technology (MIT) in Cambridge, MA, USA. Although popularly known as the 'MIT Spam Conference', the 2007 event broadened its focus to include spam, phishing and 'other cybercrimes'.

SCHEDULE

A total of 14 talks were scheduled for the one-day event. The conference grouped the talks into four tracks: invited talks (which covered blog, search engine and email spam), 'Considering the source' (with three talks covering SPF enhancements, reputation services and email header munging), 'Working the text' (which covered ground often seen at this conference: machine learning and text classification approaches to spam filtering), and 'Thinking outside the text box' (which perhaps should simply have been called 'Miscellanea', since it covered tarpitting, image spam detection and AI for responding to spammers).

As in 2006, all the talks and related papers are available for download as a disk image (in the form of an ISO file) from the conference website (http://spamconference.org). Unlike the 2006 conference, there was no real-time web cast of the event this time, but *YouTube* videos of each talk are linked from the conference website. Unfortunately, the excellent quality web cast has been replaced by very poor quality video (some without any sound at all), which makes watching the conference very trying. However, the organizer has promised that higher quality videos will be available soon.

Attendance at the conference also dropped this year, with estimates of the number of delegates ranging from 40 to 75; a far cry from the hundreds that overflowed the room back at the first conference in 2003. However, the drop in numbers is not surprising given that there are now at least two other technical conferences also covering spam: the Conference on Email and Spam (CEAS, http://ceas.cc/) and the Virus Bulletin conference (VB2007, http://www.virusbtn.com/conference/).

Nevertheless, there were some good presentations, and the broadening of the agenda gave some fresh faces a chance to present topics that have traditionally been absent from this conference's agenda.

A SPAM CHALLENGE, BLOG SPAM AND SEARCH ENGINE OPTIMIZATION

First up was Richard Segal from *IBM Research* and Gordon Cormack from the University of Waterloo. Richard talked about the upcoming 'Live Spam Challenge' that will take place as part of CEAS 2007 in August. The challenge will pit filters against each other over a 24-hour period on live spam and ham. Messages will be provided with full envelope information so that almost all spam-filtering technologies can be tested. As well as live ham and spam the system will also provide simulated user feedback throughout the day so that filters can learn from the judgements of human recipients.

Next up was one of the most interesting talks of the day (although not the winner of the can-of-spam Best Paper award): Jessica Baumgarten talking about the different types of blog spam. Unfortunately, this presentation is not available from the conference website so you'll have to make do with the *YouTube* video. [*An article by Jessica Baumgarten on blog spam is also scheduled for the June issue of Virus Bulletin - Ed.*]

After lunch, Aaron Emigh of *Six Apart* gave an unscheduled talk on the same subject (with assistance from Adam Thomason – this talk is available from the website), detailing some of the ways in which *Six Apart* deals with blog spam and showing that, once again, machine-learning filters like *CRM-114* and *DSPAM* do a good job against this particular type of spam.

Last up before the break was Amanda Watlington of *Searching For Profit*, who gave an enlightening talk about the history and state of Search Engine Optimization (SEO) and Search Marketing.

SPAM DETECTION BY HEADER/ENVELOPE INFORMATION

After a quick coffee and doughnut break the conference continued with Alberto Trevino and J.J. Ekstrom of Brigham Young University. Alberto talked about detecting spam solely by looking at header and envelope information for forged details. Just looking at HELO information they achieved a 61.8% spam detection rate with 0.33% false positives.

Looking at the validity of MAIL FROM achieved 79.3% spam detection rate with 0.53% false positives. Combining the results gave spam detection of 91.7% with 0.87% false positives. That's not as good as some machine-learning spam filter authors claim (or as test results from the TREC Spam Track show), but this technique has the important advantage that it is independent of language, obfuscation, use of images, or any other content technique spammers try to use to get around a spam filter.

IP REPUTATIONS AND TRUSTED REMAILERS

Next, Alberto Mujica (whose company *Reputation Technologies* was one of the sponsors of the event) gave a talk that described *Reputation Technologies*' service offering. In his talk he outlined the advantages of IP address reputation management.

Last up before lunch were Joseph McIsaac and Alex Pogrebnyak of *Reflexion Networks* talking about an enhancement to SPF that they term the 'Trusted Remailer' record. This record would allow a domain to publish the addresses of remailers that they trust; if the mailer is present in the record, the mail can be accepted despite the fact that the standard SPF lookup would indicate that the remailing domain was not permitted to send for a specific domain.

MODIFIED NEURONS AND SUPPORT VECTOR MACHINE FILTERS

After lunch, Alexandru Catalin Cosoi from *BitDefender* talked about combining the output of different spam filters using a modified neuron (a single perceptron) to incorporate the output of each spam filter and measure the relevance of the filter's output (the relevance can decay over time as spammers update their spam to avoid certain filter techniques). Alexandru claimed that by combining filters and using the relevance for each filter calculated by the neuron they saw an increase in spam detection accuracy and a decrease in false positives of greater than 50%. He did not, however, present any test data against any standard spam/ham set.

Next up, Ángela Blanco and Manuel Martín-Merino from the Universidad Pontificia de Salamanca talked about methods of combining Support Vector Machine (SVM) spam filters to improve accuracy. They tested a variety of techniques using a corpus of around 5,000 messages; their best result was a spam detection rate of 89.9% with a false positive rate of 1.8%. Although they showed that their technique reduced false positives significantly, it was a pity that they did not produce a comparison with simple machine-learning techniques (such as Naïve Bayes or logistic regression) on the same data set, as the figures

presented do not appear to represent an advance in the state of the art.

TARPITTING AND SMTP SLOWING

More coffee, more doughnuts and it was time for Tobias Eggendorfer from the Universität der Bundeswehr München to talk about the latest news from his SMTP tarpitting experiments. He pointed out that many of the bulk mailers have become aware of tarpitting and thus are detecting deliberate slowness and dropping connections: hence it was time to update and to use the spammers' awareness of tarpitting against them.

Tobias's basic idea is to stutter (very slowly deliver the first few bytes of a connection) and then open up the connection for full speed. To make this transparent and compatible with existing SMTP servers his implementation is a network layer 2 bridge that can achieve connection control without affecting the contents of the IP or TCP header. The stuttering will cause a spammer to drop the connection (because they think they are in a tarpit), but will not affect a legitimate sender because they'll quickly get a full speed connection once the stuttered portion is over.

By delaying each byte of the first 120 bytes of the SMTP connection transparently through the bridge by one second per byte, the total spam delivered to his test server dropped by 76.7%. In tests against real mail servers handling large amounts of ham, no false positives were observed.

This paper would have been my pick for the best paper of the conference, but the speaker after Tobias actually won the award for another sort of connection-shaping presentation.

Ken Simpson from *MailChannels* talked about his company's product (full disclosure: I am a member of *MailChannels*' technical advisory board). One nice chart from his presentation showed how connections drop off as they are slowed down: spammers drop off rapidly if they can't get a fast connection, whereas legitimate senders will hang around for minutes to get their messages delivered. He claimed that the *MailChannels*' product (whose architecture he went on to describe) drops 80–90% of spam by slowing down SMTP connections, and that the product is able to handle the incredibly high load placed on it by spammers without affecting normal email delivery. Although this was a vendor presentation the associated paper which describes the actual implementation (using Perl) is well worth reading if you are technically inclined.

IMAGE SPAM

Next up, Giorgio Fumera, Ignazio Pillai, Fabio Roli, Battista Biggio from University of Cagliari described work they are doing on detection of image spam by looking at the obfuscation techniques used by spammers trying to avoid OCR. This is exactly analogous to work done on text classification of spam, where looking at the obfuscations used by spammers is often enough to detect spam without bothering with the actual text within the message. They showed that by calculating the perimetric complexity (a measure of the complexity of a black and white image defined as the square of the length of the boundary between black and white pixels divided by the total area of black), they could detect obscured spam images.

They also mentioned that these techniques (both theirs and OCR) were often unnecessary because standard text-classifier based spam filters often have enough other text to work with (ironically, such as the random text that spammers insert to fool filters) without considering the image. And they point out that for non-obscured images OCR-ing plus text classification currently works well.

Missing from the paper and presentation was any evaluation of these techniques in the real world. They showed a number of interesting examples, but without a test against a stream of real spams and hams (both with images) it's hard to know whether these techniques would work in reality.

SPAMLET

Lastly, Kenneth P. Dallmeyer, Peter C. Nelson, Elias D. Block, Brandon R. Elvidge from University of Illinois at Chicago talked about their *Spamlet* system, which is designed to engage spammers of all types in useless conversations (such as keeping a 419 scammer emailing back and forth) to use up their resources.

Missing from the conference, but scheduled, was Nouman Azam from EME College Rawalpindi in Pakistan talking about reducing the number of features needed by a classifier and comparing term frequency, mutual information and latent semantic indexing on the Ling Spam corpus. In his paper (which is available in the downloadable conference proceedings) he determined that mutual information feature space reduction gave the best accuracy with up to 20 features.

In all, despite diminished attendance figures, the conference provided excellent material and some interesting perspectives on issues relating to spam, and I would recommend checking out the proceedings at http://spamconference.org/.

[*John Graham-Cumming will present a paper looking at past and future trends in spammer trickery, and outlining a proposed naming scheme for spammers' tricks at VB2007 in Vienna, Austria, 19–21 September. For more information and online registration see http://www.virusbtn.com/conference/.*]