DECEMBER 2005

virus BULLETIN

The International Publication on Computer Virus Prevention, Recognition and Removal

CONTENTS

2 **COMMENT**

Thoughts of mass destruction

3 **NEWS**

'Tis the season ...

Certification for adware

Calls to update the CMA

3 **VIRUS PREVALENCE TABLE**

FEATURES

4 Exploring the x64-treme heights of the Internet

7 When malware meets rootkits

11 Inside Sony's rootkit

15 **CALL FOR PAPERS**

VB2006 – Montréal

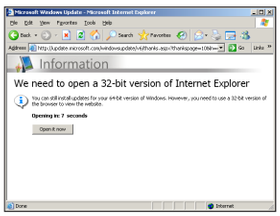
16 **COMPARATIVE REVIEW**

Windows Server 2003 Enterprise X64 version

20 **END NOTES & NEWS**

ISSN 0956-9979

IN THIS ISSUE



WHEN I'M 64

With its 64-bit *Internet Explorer*, *Microsoft* seems to have shipped a version that is relatively safe from malware, but will its lack of support for ActiveX controls, Java, Shockwave and PDFs simply drive users back to the malware-friendly 32-bit version?

page 4

THE ROOT OF THE PROBLEM

This month *VB* presents a double bill of rootkit-related articles. First, Elia Florio takes a detailed look at the rootkit technique known as 'DKOM using \Device\PhysicalMemory', then Mark Russinovich describes the rootkit discovery that ignited a firestorm of criticism for *Sony*.

pages 7 and 11

COMPARATIVE REVIEW

Dec 2005 100% VIRUS BULLETIN www.virusbtn.com

A somewhat disappointing total of nine vendors submitted their products for *VB*'s first comparative review on a 64-bit operating system.

page 16

vbSpam supplement

This month: anti-spam news & events, and we look at the possibility of a new type of spam on the horizon.

'Years after Chernobyl was released, the potential for hardware-destroying viruses has yet to be fully exploited.'

Peter Cooper, Sophos, UK

THOUGHTS OF MASS DESTRUCTION

I entered the world of anti-virus in the heady summer of '98. At the time, WM97/Class-D was proving to be a source of much amusement on the support desk. High-powered company execs and Joe Users alike were calling the support desk to find out who 'Class.Poppy' was, and why he/she/it was calling them a jerk. Perhaps more sinister was W95/CIH-10xx (aka Chernobyl), quietly infecting *Windows* computers worldwide and setting its BIOS-nuking countdown timer to 26 April 1999. WM97/Class-D was just another macro virus, but Chernobyl represented a new virus-writing era: hardware-trashing viruses.

Except it didn't. Today, years after Chernobyl was released, the potential for hardware-destroying viruses has yet to be fully exploited, despite the fact that the Chernobyl source code is available freely. There could be a number of reasons for this apparent lack of interest: the kids involved with virus-writing in the CIH era have now all grown up and found jobs, while the individuals of today's virus-writing community have a raft of prefabricated mass-mailing code available to them and constructing a virus can be a simple matter of bolting the components together. It's true that motherboard manufacturers have taken steps to minimise the impact of BIOS-trashing viruses, but where there's a will, there's a way – right? Perhaps the will has gone. Perhaps the end game for J Anonymous Virus Writer is to daub his electronic scribbles and tags all over the Internet without causing lasting damage. After all, graffiti artists don't knock down walls, they just tag them.

The prevalence of macro viruses dropped at the end of the '90s. *Microsoft Office* attachments in emails were no longer blindly being opened as often as they used to be. Could it be that computer users were finally understanding the risks associated with computing? Could people involved with IT training and security evangelism congratulate themselves on a job well done? Partly. However, a more likely explanation is that virus writers were frustrated by the limitations of the macro virus programming environment. At the same time, email servers were blocking *Microsoft Office* attachments from unknown senders. Combine the increase in user education, the decrease in interest from the virus-writing community and the non-delivery of viruses, and you have a feasible explanation for the drop in reports and new discoveries.

Chernobyl was a memory-resident, file-infecting virus. It could infect many hundreds of *Windows* files in a short space of time when the user browsed My Computer. A popular entry route for the virus was via illegally copied software and/or software cracks that the user had downloaded. But few modern viruses rely on this method to spread. Worms require a host and a network; these two things normally allow them to propel themselves from place to place. They may include file-infecting routines to spice things up, but mass destruction is not normally on the agenda. Maybe the virus writer of today has a conscience. Or maybe they're more selfish – programming software that allows them to use other people's computers for personal or monetary gain is an effective use of their time, and botnet size counts for more than a varied and successful virus-writing portfolio.

And in the future we'll see more of the same. Inevitably, some new technologies will be targeted and some as yet unused methods of distribution will be exploited. Some will be revisited from days gone by. As voice over IP gathers pace, a firewall-busting distribution network with a convenient vendor-supplied API could be harnessed for mass carnage. Maybe things will go the way of W32/Hybris and utilise newsgroups on Usenet to provide updates and plugins.

Perhaps the next big virus will be a file-infecting, massively distributed worm which auto-updates from Usenet and calls the user a jerk before trashing their BIOS. We'll see.

NEWS

Season's greetings from the VB team – clockwise from top left: Helen, Matt, Bernadette and Tom.

'TIS THE SEASON ...

The *VB* team wishes all *Virus Bulletin* readers a very happy Christmas and a prosperous and peaceful new year.

This Christmas, in lieu of sending greetings cards *VB* will make donations to two charities: The International Committee of the Red Cross (http://www.icrc.org/) and Save the Children (http://www.savethechildren.net/). Happy holidays!

CERTIFICATION FOR ADWARE

TRUSTe, a non-profit organization which certifies and monitors website privacy and email policies, has announced a new certification program for downloadable consumer software programs including adware. Under the 'Trusted Download Program', TRUSTe will create and publish a whitelist of software programs that have met a set of criteria. The whitelist will be made available to companies such as the program sponsors *Yahoo!*, *AOL*, *Computer Associates*, *CNET Networks* and *Verizon*, who will be able to consult the list before making decisions regarding advertising contracts, partner programs or distribution of software.

To gain a place on the whitelist, adware must openly disclose the types of advertising that will be displayed, reveal the personal information that will be tracked, note any user settings that it may alter, and must obtain opt-in consent for the download. In addition, programs must provide easy-to-follow uninstallation instructions and the advertisements displayed must carry the name of the adware program. The Beta launch of the Program is scheduled for early 2006.

CALLS TO UPDATE THE CMA

An MP is calling for support of his proposals to update the UK's Computer Misuse Act (CMA) following the collapse of the trial of a DoS attacker last month. The case was dismissed when the judge ruled that executing a DoS attack did not contravene the CMA. The relatively antiquated Act was drafted in 1990, and MP Tom Harris believes an overhaul is long overdue. Harris has called on MPs to approve his Private Members Bill amending the Act, which seeks to criminalize all means of interference with a computer system and creates an offence of DoS attacks. The Bill also increases the penalty for hacking offences from six months to two years. It is due for its second reading in the House of Commons on 17 March.

Prevalence Table – October 2005

Virus	Type	Incidents	Reports
Win32/Mytob	File	169,638	53.21%
Win32/Netsky	File	115,268	36.16%
Win32/Mydoom	File	16,130	5.06%
Win32/Bagle	File	5,931	1.86%
Win32/Zafi	File	2,356	0.74%
Win32/Sdbot	File	2,211	0.69%
Win32/Funlove	File	1,211	0.38%
Win32/Lovgate	File	940	0.29%
Win32/Sober	File	866	0.27%
Win32/Mabutu	File	627	0.20%
Win32/Bugbear	File	378	0.12%
Win32/Agobot	File	362	0.11%
Win32/Valla	File	320	0.10%
Win32/Klez	File	265	0.08%
Win32/Gibe	File	218	0.07%
Win32/Pate	File	210	0.07%
Win32/Mimail	File	185	0.06%
Win32/SirCam	File	177	0.06%
Win32/Dumaru	File	175	0.05%
Win32/Bagz	File	113	0.04%
Win32/Maslan	File	99	0.03%
Win32/Swen	File	93	0.03%
Win32/Bobax	File	90	0.03%
Win32/Reatle	File	83	0.03%
Win32/Elkern	File	79	0.02%
Win95/Tenrobot	File	70	0.02%
Win32/MyWife	File	69	0.02%
Win32/Mota	File	66	0.02%
Redlof	Script	65	0.02%
Win32/Fizzer	File	37	0.01%
Win32/Yaha	File	37	0.01%
Win32/Gael	File	34	0.01%
Others[1]		399	0.13%
Total		318,802	100%

[1]The Prevalence Table includes a total of 399 reports across 65 further viruses. Readers are reminded that a complete listing is posted at http://www.virusbtn.com/Prevalence/.

FEATURE 1

EXPLORING THE X64-TREME HEIGHTS OF THE INTERNET

Costin Raiu
Kaspersky Lab, Romania

Having released two major *Windows* versions for the x64 architecture (also known as 'AMD64'), *Microsoft* has opened the door to inexpensive 64-bit computing for just about everybody. At the 2005 Virus Bulletin Conference I presented a paper [1] on the x64 architecture, detailing how known 32-bit viruses and rootkits interact with it.

There were a number of questions from the audience at the end of the presentation, but the most interesting one was a question posed by a researcher from *Symantec*'s European AntiVirus Research Centre. It related to *Internet Explorer* in *Windows x64* and how ActiveX objects and BHO (Browser Helper Objects) – which until now have been exclusively 32-bit – work (or don't work) in the *Windows x64* environment. The significance of this, of course, is that there is a great plethora of spyware and adware which installs from the Internet via *IE* vulnerabilities, through the use of ActiveX technology or Java applications.

This article is written in response to that question.

MEET INTERNET EXPLORER X64

Since all the system applications in *Windows x64* are 64-bit PE files, one would assume that *Internet Explorer* has also been compiled as a 64-bit native application. The assumption is correct – a 64-bit *Internet Explorer* (version 6.0.3790) is built into the most recent *Windows XP* release for the x64 platform. Its 'About' dialog is shown in Figure 1.

However, what may not be so obvious is that, alongside the 64-bit *Internet Explorer* there is also a 32-bit version of *IE* available from the Start menu (see Figure 2). When run, the 32-bit *Internet Explorer* looks identical to its 64-bit twin, but a quick check in the Task Manager shows that it is indeed a 32-bit process.

So it seems that the *Windows* developers have decided to offer the user the choice of which version of *Internet Explorer* to use. Although this may sound a little strange, there is a very good reason: compatibility.

In *Windows x64*, the 64-bit applications can make use of IPC to call older 32-bit code, just like 32-bit applications can call 16-bit DLLs in Win32 through the use of thunking. This means that, at least in theory, any 64-bit application could be made to use a 32-bit DLL. However, calling 32-bit code from a 64-bit application does not come without drawbacks, speed and security being the main concerns.

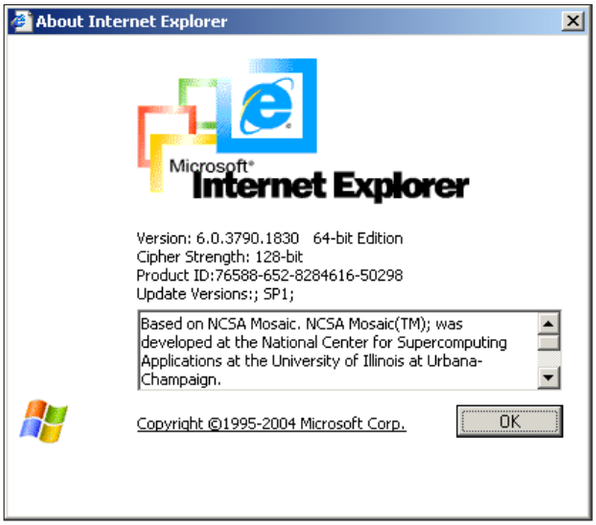


Figure 1: Internet Explorer 64-bit Edition.

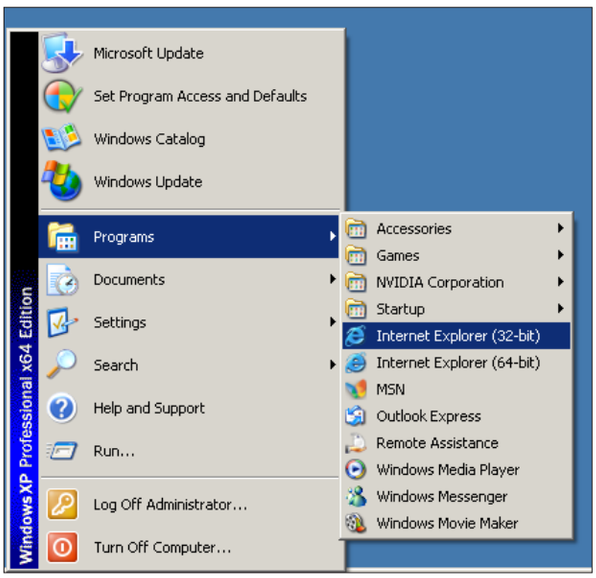


Figure 2: A 32-bit version of the same product is available from the Start menu.

Notwithstanding these concerns, in theory, the 64-bit *Internet Explorer* could have made use of 32-bit ActiveX objects and plug-ins. So why did *Microsoft* choose to ship a 32-bit version of *Internet Explorer* with *Windows XP x64*?

One simple explanation is that to make 32-bit ActiveX objects work from the 64-bit *IE*, *IE* would have to open a 32-bit VM, load the 32-bit COM object in there and communicate with it. The technical effort required would not be trivial, but for a company of *Microsoft*'s size, this should not be a serious problem.

Another possibility is that having 32-bit ActiveX objects enabled in the 64-bit *Internet Explorer* opens the door to a large set of security threats that have been plaguing the Internet for years. It also requires the coding of a special application gateway: a 64-bit wrapper for ActiveX objects that lessens the separation between the 64-bit and 32-bit worlds in *Windows x64*. This is important because, for example, 64-bit and 32-bit applications 'see' different versions of the registry and can of course have different types of configuration settings and therefore behave differently.

Whichever of the explanations (or other unexplored possibilities) is correct, *Windows x64* includes two versions of *IE*. Let's see how they handle ActiveX controls.

THE ACTIVEX TECHNOLOGY

ActiveX is a generic name for a set of technologies which have been developed with the purpose of sharing information between applications. ActiveX is related to COM and OLE, but most of the time ActiveX is used to denote 'ActiveX controls'. In this case it is the technology designed to be called from a web browser.

Many years ago, when ActiveX was made available in *Internet Explorer*, it was considered a direct competitor for Java. However, *Microsoft*'s *IE* was the only browser on the market that supported it. One of the main arguments against ActiveX was that it was based on x86 executable code, making it very hard to operate on other CPUs such as *PowerPC*s or *Sparc*s. As a result, companies writing true multiplatform applications would be limited to running them through *Microsoft*'s browser, and only on *Intel*-compatible PCs.

In fact, *Microsoft* even shot itself in the foot with ActiveX, as running *Internet Explorer* doesn't necessarily mean that you have an environment which supports ActiveX! For instance, the MacOS *Internet Explorer* (most recent version: 5.2.3) is unable to load and run ActiveX controls. So, while Java meant that you have a variety of systems and hardware architectures on which to run your applications, ActiveX meant that your code would run on *Intel* x86 32-bit CPUs under *Windows*.

With the arrival of *Windows x64*, ActiveX developers have one more platform to worry about – 64-bit *Internet Explorer* ActiveX controls.

PECULIARITIES OF 32-BIT AND 64-BIT INTERNET EXPLORER IN WINDOWS X64

The 64-bit *Internet Explorer* from *Windows x64* is located in the 'Program Files' folder, separated from the 32-bit version. It supports plug-ins and, as one might have expected, it is able to load and run 64-bit ActiveX controls.

However, it is interesting to note that *Windows x64* defaults to the 32-bit *Internet Explorer* in most cases. The 32-bit *IE* is called when clicking the *IE* shortcut on the Taskbar, from the 'Windows Update' Start Menu item and from the Desktop *IE* Shortcut. Yet, when an Internet URL is requested from the *Windows Explorer* – which is a 64-bit application itself – the 64-bit *IE* is used.

The behaviour of 'Windows Update' is also interesting under *Windows x64*. As mentioned, the Start Menu item for 'Windows Update' launches the 32-bit *Internet Explorer,* which loads all the update controls – which are 32-bit applications themselves. However, if you open the Windows Update website in the 64-bit *Internet Explorer*, you get the message shown in Figure 3.

Unfortunately, I do not have an *Itanium* IA64 machine handy to test whether the Windows Update has a native IA64 ActiveX control, or whether the update is run as emulated 32-bit code, but this is definitely worth trying.

Continuing with the peculiarities of the 32-bit and 64-bit versions of *IE* from *Windows x64*, it is worth noting that they share a couple of settings, such as using the same Temporary Internet Files folder (which defaults to 'Documents and Settings\%Username%\Local Settings\Temporary Internet Files'), the same History folder and the same Favorites folder.

In version *IE 6 SP2*, *Microsoft* introduced a very useful feature to manage the *IE* Add-ons. It can be found in the Tools menu: 'Tools->Manage Add-ons'. The Add-ons Manager can be used to disable or re-enable ActiveX controls or Browser Extensions, but unfortunately it can't be used to uninstall any of these. This gives the user a better idea of the extensions currently used by *IE*, as well as providing a moderate degree of control over them.

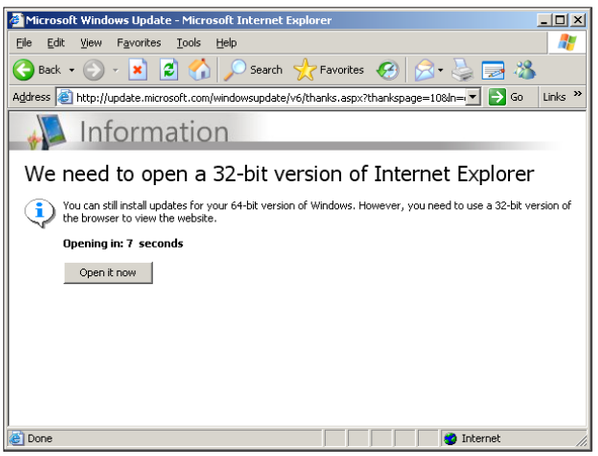


Figure 3: Opening the Windows Update website in 64-bit Internet Explorer.

64-BIT IE PLUG-INS

Neither of the versions of *IE* from *Windows x64* supports Java by default, so one has to download and install Java support separately. Of course, there is a 32-bit JRE in the most recent release from *Sun* (*J2SE Runtime Environment 5.0* Update 5), but there is no *Windows* AMD64 support listed on *Sun*'s website. There is an AMD64 version of the *J2SE Runtime Environment 5.0* in Update 4, which can be obtained from *Sun*. However, it doesn't seem to work in *Windows XP x64*. It installs without error messages, but *IE* does not include it in the list of Add-ons loaded, and Java applets do not work when called from a web page.

Additionally, there is no *Windows x64 Flash Player* from *Macromedia*, no *Acrobat Reader x64* from *Adobe* and no direct in-browser *QuickTime* support from *Apple*. In fact, at the time of writing this article, I have been unable to find any 64-bit *IE* plug-in, or third-party 64-bit ActiveX control.

This means that the 64-bit *Internet Explorer* experience does not (yet) support Java applets, Shockwave or PDFs, which in practice limits its usefulness to the point where many users will have no reason to use it at all. Except of course, if no support for the above-mentioned types of content is what they want in the first place.

MALWARE ATTACKS

As a result of the inability of the 64-bit *IE* to call 32-bit x86 code, current 32-bit malware attacks are greatly limited. In practice, none of the current methods used to deliver malware through web pages work, except maybe for direct download and execution by the user.

Not surprisingly, the same cannot be said for the 32-bit *Internet Explorer* from *Windows x64*. I have tested a range of malware that installs itself from various web pages, including but not limited to (all *KAV* names):

- Trojan-Downloader.Win32.IstBar.jm
- Trojan-Downloader.Win32.IstBar.kq
- Trojan-Downloader.IstBar.ij
- Trojan-Downloader.Win32.Dyfuca.ei

These are all delivered from web pages through the use of various *IE* exploits (which are already patched in the *IE6* for *Windows XP x64* so they don't work) and through the use of ActiveX controls. Some of these use Win32 kernel mode rootkits to hide themselves (which do not work in *Windows x64*) and some of them rely on user-mode rootkits, which are effective against other Win32 applications in *Windows XP x64*, but not against 64-bit applications.

The bad news is that not only they do infect the system successfully (although they all require the user's confirmation to install), but they also operate without glitches under *WOW64*. They survive reboot even if the BHOs are disabled from *IE* prior to restarting, and they are not easy to get rid of without a full-blown 64-bit anti-virus scanner.

In fact, without a 64-bit anti-virus scanner (and unfortunately, the offerings are still very limited in this area) getting rid of 32-bit malware on *Windows x64* is not a straightforward task for the inexperienced user. Of course, an experienced user could always kill all 32-bit processes in Task Manager, and then use a plain 32-bit scanner to disinfect or delete all the infected files, but the separation between 32-bit processes in *Windows x64* makes it quite difficult for a 32-bit AV scanner to remove some of the above-mentioned malware while they are active.

CONCLUSIONS

With *Windows x64*, *Microsoft* seems finally to have shipped a version of *IE* which is pretty much safe from most of today's *IE*-oriented malware attacks – I'm talking about the 64-bit *IE*, of course. The bad news is that this isn't thanks to a breakthrough in security on *Microsoft*'s part, but comes as a side effect of the total absence of support for Java, 32-bit ActiveX controls, *Shockwave* content and PDFs. And, thanks to this lack of support, most users will simply use the 32-bit *Internet Explorer*, which as we've seen, is just as malware-friendly as any other version running on *Windows XP 32-bit*.

However, with an up-to-date 64-bit AV scanner with the O/A component activated, most users should have no problem dealing with 32-bit malware – in fact, 32-bit malware should be easier to deal with because of the good separation between the 64-bit and 32-bit worlds in *Windows x64*. In this case, users of *Windows x64* will have fewer problems with malware than their 32-bit counterparts.

That is until 64-bit malware starts to become popular, of course.

REFERENCES

[1] Costin Raiu, '"Enhanced" virus protection', *Proceedings of the Virus Bulletin International Conference 2005*, pp.131–138.

[2] 'Microsoft: List of limitations in 64-Bit Windows', http://support.microsoft.com/kb/282423.

[3] 'Microsoft: Differences between the 32-bit and 64-bit versions of Internet Explorer that are included in the x64-based versions of Windows Server 2003 and in Windows XP Professional x64 Edition', http://support.microsoft.com/kb/896457.

FEATURE 2

WHEN MALWARE MEETS ROOTKITS

Elia Florio
Symantec Security Response, Ireland

There was a time when *Windows* rootkits were just stand-alone applications, but today it's very common to find advanced rootkit technologies used in worms and Trojans – and sometimes even in non-malicious programs. Although *Windows* rootkits were introduced only a few years ago, the number of programs that currently use stealth technology, or that will use it in the future, is growing very quickly, sometimes with unexpected consequences.

THE ART OF HIDING EXPRESSED IN MANY FORMS

This article will not cover all the techniques of rootkits, since the topic is huge. For information on rootkits and how they work on *Windows* operating systems, refer to [1].

This article deals only with a specific rootkit technique known as 'DKOM using \Device\PhysicalMemory'. This technique was observed recently in the worm W32/Fanbot.A@mm [2], which spread worldwide in October 2005. The article will also present some data on rootkit usage in malicious threats.

Rootkits are usually divided into two categories: user-mode rootkits that work in Ring3 mode, and kernel-mode rootkits that operate in Ring0. The latter represents a more sophisticated piece of code, which requires a lot of programming knowledge and familiarity with the *Windows* kernel.

Kernel-mode techniques are very powerful and the most advanced rootkits are able to subvert the *Windows* kernel [3] and hide files, folders, registry keys, ports and processes. This type of rootkit needs to operate as a system driver to manipulate the kernel because this interaction requires Ring0 privileges, which are not available for normal executables in userland space.

The major drawback of this implementation is that the rootkit always comes with two different binaries (one SYS driver and one EXE that installs the driver) and this fact raises some barriers to the practical integration of this type of threat into real applications. Even if the SYS driver can hide everything (including itself), it needs to keep static structures installed in kernel memory which can be detected [4]. Moreover, the installation process requires interaction with the *Windows* Service Control Manager (SCM), or alternatively the undocumented API ZwSetSystemInformation. Both methods can create some evidence of the threat's presence or can be blocked during the installation phase.

GHOST PROCESS IN THE SYSTEM

For these reasons, the next generation of rootkits started to approach the *Windows* kernel in a different way, avoiding the need for a SYS driver and system hooks. This goal is achieved by mixing the idea introduced by the FU rootkit (known as DKOM, Direct Kernel Object Manipulation) with another technique that involves the manipulation of the \Device\PhysicalMemory object and does not require any additional driver. The method of 'playing' with the physical memory object was imported from the *Linux* world, where another (in)famous rootkit known as 'SucKIT' [5] is gaining a lot of popularity.

DKOM rootkits are able to manipulate kernel structures and can hide processes and ports, change privileges, and fool the *Windows* event viewer without many problems. This type of rootkit hides processes by manipulating the list of active processes of the operating system, changing data inside the EPROCESS structures. This method is well documented and was first implemented by the FU rootkit [6].

Essentially, the *Windows* operating system maintains two different lists of all process and thread information (PID, name, token, etc.). Every process has an associated EPROCESS structure, which is linked to the previous and the following process (double-linked list) using some pointers. Figure 1 shows, with a simplified diagram, how EPROCESS structures are interconnected.

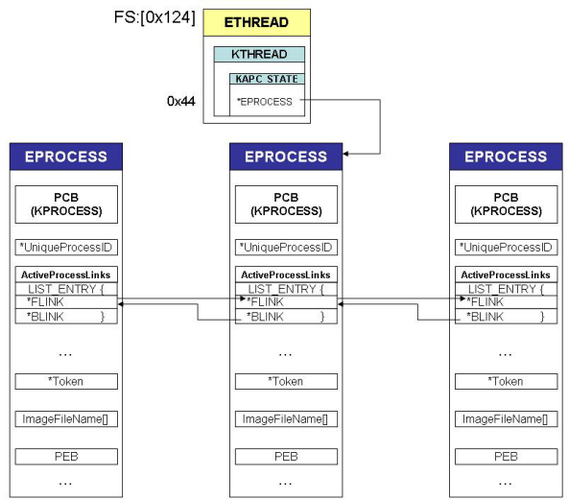


Figure 1: Windows EPROCESS structures are connected to each other by a double-linked list.

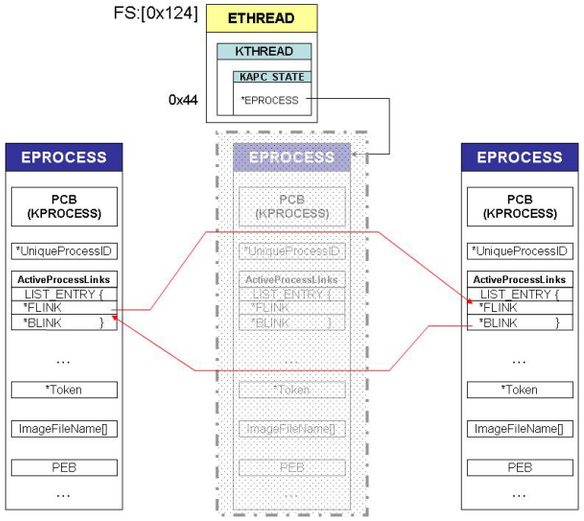


Figure 2: To hide a process the DKOM rootkit simply unlinks it from the list, linking its previous process with the next one. It's just a swap of a few pointers.

However, many people don't realise that processes don't run; only threads run. The *Windows* operating system uses a pre-emptive, priority-based, round-robin method of scheduling threads, swapping the active status from one thread to another (process structures are not involved in the switch).

Considering this fact, DKOM rootkits exploit a very simple trick: they unlink their own EPROCESS from this list, connecting the pointers of the previous and of the next EPROCESS in a way that will skip the 'ghost' process.

With this simple change, a process becomes invisible to the task manager and other common process manager tools, but it still runs in the system as all its threads are still active. Only advanced tools (e.g. KProcCheck [7]) can detect the presence of the hidden process by traversing the handle table list or the scheduler thread list.

This kind of threat (DKOM rootkit that uses \Device\PhysicalMemory) is quite hard to code because it requires the following abilities:

1. The ability to obtain read/write access to the \Device\PhysicalMemory object.
2. The ability to manipulate the EPROCESS/ETHREAD structure correctly (these structures differ greatly between *Windows 2000*, *XP* and *2003*).
3. The ability to locate the 'System' process in kernel memory and patch it.
4. The ability to translate the virtual address of a process to a physical address in memory.

While there have been good examples of the first three steps [8] in the past, the last step is the most difficult as the *Windows* addressing scheme is based on a complex layer of multiple arrays. Contiguous virtual addresses of a process may have different physical addresses mapped into kernel memory [9].

WORMS USING \DEVICE\PHYSICALMEMORY

It was surprising to find a practical (and well-written) implementation of this rootkit technique inside the W32/Fanbot.A@mm code. W32/Fanbot.A is not the only worm that uses the DKOM and \Device\PhysicalMemory technique. The first worm that tried to achieve this was W32/Myfip.H. However, the routine observed in this worm was a little buggy and did not work well under *XP* and *2003* systems as it used a simplified memory model (the trick introduced in [8]) to map logical addresses to physical addresses.

W32/Myfip.H tried to 'emulate' the kernel API MmGetPhysicalAddress by checking if the virtual address was in the range (0x80000000 – 0xA0000000) and applying to it an AND mask of 0x1FFFF000. However, MmGetPhysicalAddress changes a lot from *Windows 2000* to *XP*, so the correct way to translate the virtual address is to use the page tables of the specific process that owns the virtual address to be translated.

Instead, W32/Fanbot.A implements a good algorithm for address translation that considers the Page Directory and the Page Table (including tests for large pages). It also follows all the basic memory management rules: it extracts PDindex

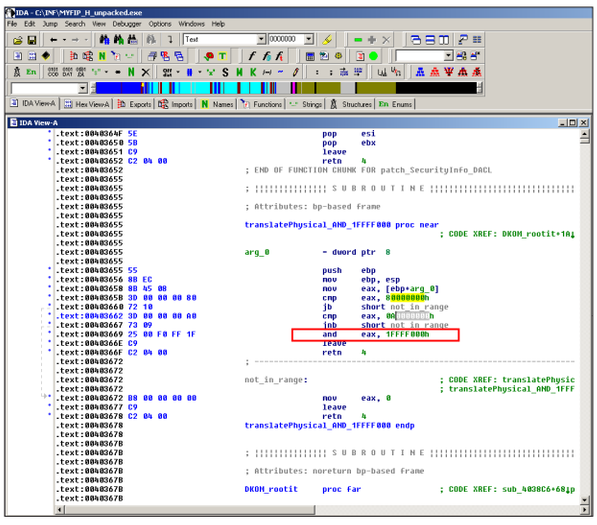


Figure 3: The Myfip.H variant implemented a DKOM routine patching physical memory object, but it uses a simplified translation algorithm for addresses.

from the virtual address, gets the correct PDE, locates the corresponding PTE, and finally calculates the correct physical address. The only limitation of the W32/Fanbot.A code is that it does not work on *Windows* versions with PAE (Page Address Extension), because it makes the assumption of four-byte entries for PD and PT.

A CLOSER LOOK AT THE ROOTKIT CODE USED BY FANBOT

W32/Fanbot.A@mm is a worm that has all the typical mass-mailing techniques. This variant comes packed with *NsPack* and installs itself as a service. It can spread by email, copy itself into P2P folders, and exploit the universal plug-and-play vulnerability (MS05-039).

Once unpacked (276 KB of code), it's possible to locate the DKOM routine by searching for the unicode string '\Device\PhysicalMemory' and tracing its reference back to the virtual address 0x40F8A5, where the rootkit code begins. The nice thing (for malware writers) is that the rootkit routine of the worm is written in a modular way so that it can easily be extracted and reused in any other malware.

First, the worm loads the NTDLL.DLL library and gets the APIs that are necessary to operate (RtlInitUnicodeString and ZwOpenSection).

Next, it checks the OS version and uses an interesting technique to locate the PDB (Page Directory Base) of the 'System' process. DKOM rootkits need to locate the System process in order to get its PDB (which is necessary for physical address translation). For example, the FU rootkit tries to locate System by iterating all the EPROCESS structures and looking for the 'System' string in the name. Other rootkits find the System process by checking UniqueProcessID, because on *Microsoft* systems the following assumption is usually true:

- *Windows NT / 2000* => 'System' PID = 8
- *Windows XP / 2003* => 'System' PID = 4

However, W32/Fanbot.A uses a completely different method: it does not scan for a string or PID – it only checks the OS version and locates the PDB of the System process directly using one of the following offsets (as explained in [10]):

- *Windows 2000* => 'System' PDB = 0x30000
- *Windows XP* => 'System' PDB = 0x39000

At this stage the worm is ready to open '\Device\PhysicalMemory' using ZwOpenSection. If it fails (usually because the current user has no rights to manipulate this object) then it uses the trick (described by Crazylord in [8]) of changing ACLs (adding Read/Write permissions) for the physical memory.

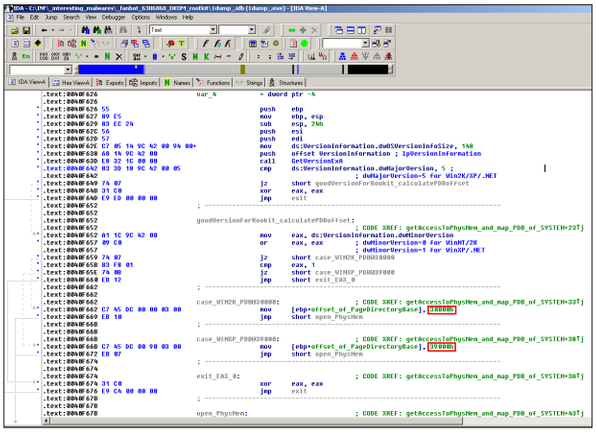


Figure 4: The rootkit routine of W32/Fanbot.A worm is able to work under Windows 2000 and XP, as it knows all the correct offsets of several kernel structures.

Once the worm has located the System page directory, it reads the PDB from memory and keeps a copy of it for all the address translations. The rootkit routine follows this procedure:

1. Locate the current running ETHREAD structure at 0xFFDFF124 (FS:0x124).
2. From ETHREAD jump to EPROCESS, using the pointer at offset 0x44 of the structure.
3. Read FLINK and BLINK from ActiveProcessLinks of the current EPROCESS structure (these offsets change from *2000* to *XP*).
4. Unlink the current EPROCESS from the ActiveProcessLinks list by connecting the previous process with the next one (just a swap of a few DWORDS!).

The Fanbot worm works under *Windows 2000* and *XP* because the author implemented all the necessary checks for different OS versions, and because it uses the right offsets to handle the EPROCESS structures correctly, according to the following table:

	Windows 2000	**Windows XP**	**Windows 2003**
PID offset	0x94	0x9C	0x84
FLINK offset	0xA0	0x88	0x88
BLINK offset	0xA4	0x8C	0x8C

Table 1: Some important offsets of the EPROCESS structure that change for different Windows versions.

After the end of the rootkit routine, the worm executable is completely hidden and disappears from the process list.

ROOTKIT TECHNOLOGIES IN THE WILD

The recent *Sony* digital rights management case is evidence of how mature rootkit technology has become a commercial entity ([11] and p.11). This rootkit has caused general consumer uproar as can be seen simply on *Amazon*'s feedback pages for several *Sony* CDs that ship with the rootkit (see http://www.amazon.com/).

But if rootkits have gained this much popularity in the software industry, what's been happening in the 'malware industry'? A process of rootkit integration has already started and many examples of different rootkit techniques can be seen in Trojans, worms, and now also in spyware and adware programs. Malware writers have learned the lesson and they know that the hardest enemy to fight is the one that nobody can see!

NAME	THREAT CATEGORY			ROOTKIT CHARACTERISTICS				
	Worm/ Virus	Backdoor/ Trojan	Adware/ Spyware	DLL / IAT hooking	SDT / IDT hooking	DKOM	Use SYS driver	Use "PhysicalMemory"
Adware/Elitebar			X	X				
Adware/CommonName			X		X		X	
Spyware/ISearch			X		X		X	
Spyware/ElpowKeylogger			X		X		X	
Spyware/Apropos.C			X	X	X		X	
Backdoor/Graybird (fam.)		X		X	X		X	
Backdoor/Haxdoor (fam.)		X			X		X	
Backdoor/Darkmoon (fam.)		X			X		X	
Backdoor/Berbew (fam.)		X		X	X		X	
Backdoor/Ryejet (fam.)		X			X		X	
Trojan/Drivus		X			X		X	
PWSteal/Raidys		X			X		X	
W32/Spybot.NLX	X				X		X	
W32/Theals.A@mm	X			X				
W32/Tdiserv.A	X				X		X	
W32/Mytob.AR@mm	X				X		X	
W32/Loxbot.A@mm	X				X		X	
W32/Myfip.H@mm	X					X		X
W32/Fanbot.A@mm	X					X		X

Table 2: List of malware and security risks that use rootkit techniques to hide files, processes or registry keys. In some cases it is possible to observe completely different rootkit techniques used by variants of the same family (e.g. Backdoor/Graybird). Some malware, like W32/Loxbot.A@mm, contain a modified copy of FU rootkit (msdirectx.sys) embedded in their code.

REFERENCES

[1] Patrick Runald, 'The trouble with rootkits', *Virus Bulletin*, September 2005, p.4.

[2] Description of W32/Fanbot.A@mm, http://securityresponse.symantec.com/avcenter/venc/data/w32.fanbot.a@mm.html.

[3] Greg Hoglund and Jamie Butler, *Rootkits: Subverting the Windows Kernel*, Addison-Wesley Professional, 2005.

[4] 'modGREPER', a hidden module detector created by Joanna Rutkowska, http://invisiblethings.org/tools/modGREPER/modGREPER-0.2-bin.zip.

[5] Sd and Devik, 'Linux on-the-fly kernel patching without LKM', *Phrack* #58, Article 7, http://www.phrack.org/show.php?p=58&a=7.

[6] FU Rootkit, http://www.rootkit.com/project.php?id=12.

[7] 'Win2K Kernel Hidden Process/Module Checker' KprocCheck by SIG^2 http://www.security.org.sg/code/kproccheck.html.

[8] Crazylord, 'Playing with Windows /dev/(k)mem', *Phrack* #58, Article 16, http://www.phrack.org/show.php?p=59&a=16.

[9] Pankaj Garg, 'Windows Memory Management', http://www.intellectualheaven.com/Articles/WinMM.pdf.

[10] Mark Russinovich and David Solomon, *Microsoft Windows Internals*, Fourth Edition, Microsoft Press, 2004.

[11] XCP (eXtended Copy Protection), the digital audio protection used by *Sony* which makes use of rootkit technology http://www.xcp-aurora.com/.

FEATURE 3

INSIDE SONY'S ROOTKIT

Mark Russinovich
Sysinternals, USA

In late October 2005, as I was performing scans of my computer systems with a test version of *RootkitRevealer* – a rootkit detection tool I had co-authored with Bryce Cogswell – I was stunned to see evidence of the presence of a rootkit on one of my computers.

RootkitRevealer displayed a number of cloaked Registry keys and files and a hidden directory – Figure 1 shows the results. The cloaked objects were not connected in any obvious way to software that I had installed, so I launched an investigation using several tools. Eventually I came to the conclusion that the rootkit had been installed when I had accepted a EULA presented to me by the autorun program on a *Sony* CD I had purchased: *Get Right With the Man*, by Van Zant.

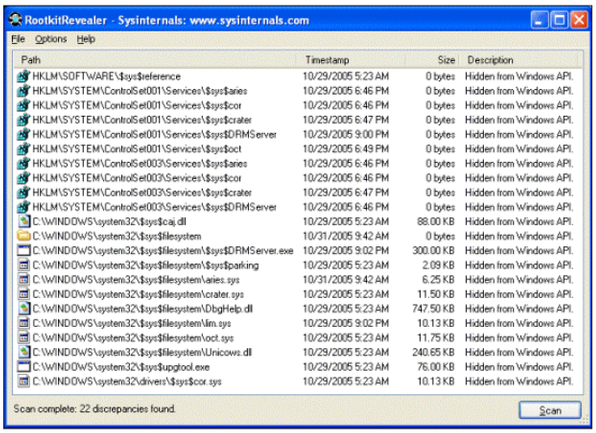


Figure 1: RootkitRevealer results on a system with XCP installed.

I documented the findings in my blog (http://www.sysinternals.com/blog), and a firestorm of criticism aimed at *Sony* ensued. Less than 24 hours after publishing my findings hundreds of comments had been posted to the blog and the story was picked up by *Slashdot.org*.

The furor centred on the fact that *Sony*'s Digital Rights Management (DRM) software, which it had licensed from UK-based *First4Internet*, used the rootkit's cloaking to hide its presence from computer users without asking them explicitly for consent to such behaviour, or even noting it in the EULA.

As the story unfolded more problems came to light, including the 'phone home' behaviour of the player that ships on the CD, possible use of LGPL software in the player, and security problems in the ActiveX control that *Sony* provided originally as its uninstaller. Eventually, *Sony* discontinued the production of CDs using *First4Internet*'s 'XCP' DRM technology, recalled all of the CDs containing XCP, offered existing customers an exchange for non-XCP CDs, and provided a stand-alone executable uninstaller.

There are many interesting angles to this story, but in this article I focus on the XCP rootkit's implementation and discuss the use of rootkits in commercial software.

SYSTEM CALL HOOKING

Aries.sys is the device driver that implements the rootkit functionality of the *First4Internet* DRM software. The CD installation software starts and loads the driver when a user accepts its EULA and configures the driver as an auto-start driver, which means that it loads early each time *Windows* boots.

Under most circumstances, *Windows* presents end users with an acceptance dialog box before installing unsigned drivers. Unsigned drivers have not been subject to reliability testing by *Microsoft*'s Windows Hardware Quality Laboratories (WHQL), so *Microsoft* cannot know what level of testing the drivers have undergone.

Neither Aries nor the other drivers included in the XCP are signed, but *Windows* performs signature checks only when it installs drivers via its Plug and Play system. Since the CD installation configures the XCP hardware-related drivers manually and starts all the XCP drivers, including Aries, using the Windows Service Control Manager's StartService API, no check occurs and no warning is presented. (*Microsoft* is considering implementing a check for a signature in all driver load paths in *Windows Vista.*)

The Aries driver relies on system call hooking, a technique I pioneered with Bryce Cogswell in 1996 with the first

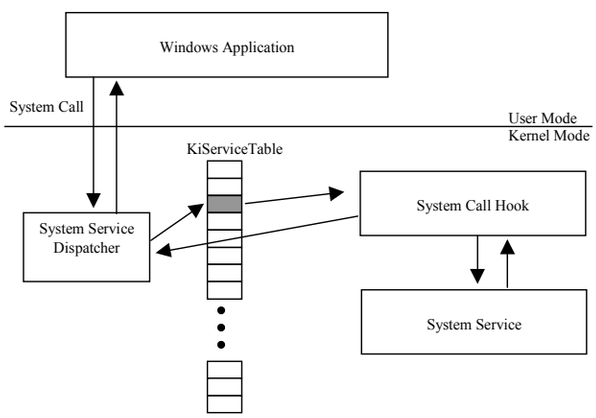


Figure 2: System call hook control flow.

implementation of the Regmon registry monitoring application. When a user-mode process invokes a kernel-mode system service it does so by loading the index number of the service into a processor register and then transitioning to the kernel-mode system service dispatcher, by executing either a software interrupt or a processor-supported system call instruction. The system service dispatcher locates the target kernel-mode service by calling indirectly through the specified index in the system service table, which is identified internally as KiServiceTable (see *Windows Internals* by Russinovich and Solomon for more information).

A system service is hooked when a driver replaces the function pointer in KiServiceTable for the service with a pointer to its own function. Subsequent calls to the service route to the hooking driver, which can examine and manipulate input parameters, invoke the original routine, manipulate output parameters, or even process the service without the use of the original service routine. Figure 2 depicts a system call hook.

The list of functions that Aries hooks when it initializes, along with their *Windows* API wrappers that export the services to user-mode, are shown in Table 1. Modern rootkits can cloak operating system objects ranging from TCP/IP ports to *Windows* services, but a cursory examination of the system services that Aries hooks reveals that it cloaks only file system, Registry, and objects returned by ZwQuerySystemInformation.

System service	Windows API wrapper	Description
ZwCreateFile	CreateFile	Opens a file or directory
ZwQueryDirectoryFile	FindFirstFile, FindNextFile	Lists directory contents
ZwOpenKey	RegOpenKeyEx	Opens a Registry key
ZwEnumerateKey	RegEnumKeyEx	Lists a key's subkeys
ZwQuerySystemInformation		Queries system information

Table 1: System services hooked by Aries.

When I performed my initial investigation of the rootkit I looked for evidence of system-call hooking by examining the system service table using local kernel debugging. Local kernel debugging describes the use of a kernel debugger running on a system to examine the kernel code and data of the same system.

LiveKd, a tool I released on the CD accompanying *Inside Windows 2000* and now available from *Sysinternals*, provides local kernel debugging capability for the standard *Microsoft* kernel debuggers, Kd and Windbg, on *Windows NT 4.0* and higher, and *Microsoft* added local kernel debugging support in the *Windows XP* kernel. Because all system services reside within the core operating system image file, Ntoskrnl.exe, hooking is visible as offsets in the system service table that fall outside this image. Figure 3 shows two easily-identifiable hooks inserted by Aries.

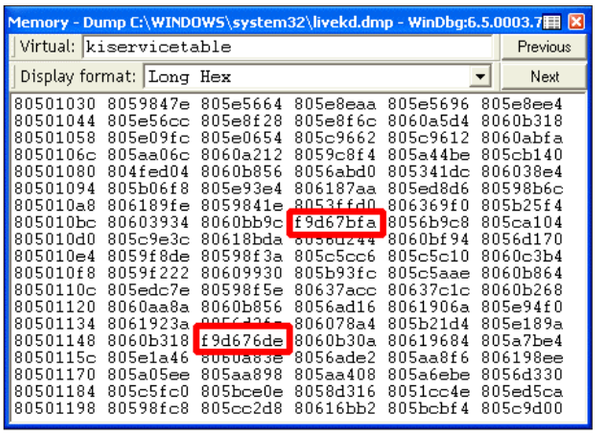


Figure 3: Evidence of Aries hooks.

Until a few years ago we made the source code to Regmon available publicly, which led to the use of our hooking functions and support routines in the NTRootkit example that's published on http://www.rootkit.com/. The structure of the code in Aries indicates that it's likely to be derived from NTRootkit code.

As an aside, system call hooking is prevented by *Windows 64-bit Editions* for the x64 platform through a technology *Microsoft* calls 'patch guard'. Patch guard monitors the system service table and other system structures that are commonly altered by rootkits, and crashes the system when it detects a modification. The alternative to system call hooking on these platforms for Registry operations is the Registry callback interface that the 5.2 version of the kernel introduced; file system filtering is the kernel framework available for monitoring and altering file system operations.

HOOK IMPLEMENTATION

Like most rootkits, Aries cloaks objects for all processes except for those that it considers privileged enough to see an accurate view of the system. The first step of each of the hooks, therefore, is to query the name of the process executing the hooked service.

Windows stores the image name of a process in the executive process block (EPROCESS). To determine the

version-dependent offset of the process name field in the EPROCESS structure the Aries initialization code uses Regmon-based code to search for the name 'System' in the EPROCESS of the System process, which is the process in which it executes.

If the process name is prefixed with the string 'sys', Aries allows execution to proceed unaltered by executing the system service function it hooked and returning the result. The Sony DRM software includes one process, sysDRMService.exe, that receives an unfiltered view of the system.

When a process does not have a privileged name the hook functions simply filter objects that also have names with the 'sys' prefix. The hook for one of the two file-related functions hooked by Aries, ZwCreateFile, returns the *Windows* native error code, STATUS_OBJECT_NOT_FOUND, which translates to the user-mode ERROR_PATH_NOT_FOUND error, for attempted opens of files and directories with such names.

The other file-related service, ZwQueryDirectoryFile, returns a list of child files and directories of a particular directory that match specified search criteria and the Aries hook removes entries with the 'sys' prefix.

The ZwRegEnumerateKey hook behaves in a manner similar to that of ZwQueryDirectoryFile, stripping sys-prefixed keys from the result buffer returned by the underlying service. ZwOpenKey's hook is different, however, because it does not modify the functionality of ZwOpenKey.

It appears that the developer originally intended to model the ZwOpenKey hook on that of ZwCreateFile, but realized that doing so would prevent the Service Control Manager, Services.exe, from opening the keys corresponding to DRM-related services and drivers, such as sysDRMServer, and therefore prevent those services and drivers from loading. The reason that the hook remains is likely to be an oversight.

The final hook function, that for ZwQuerySystemInformation, cares about only one particular type of system query: SystemProcessInformation. This is the query that process diagnostic tools like Task Manager use to obtain a list of active processes. Operating the same way as the other hooks, ZwQuerySystemInformation filters processes that have names starting with 'sys' from the list returned by the kernel function.

The effect of the Aries hooks is to hide the presence of directories, Registry keys and processes that have the 'sys' prefix from any process that doesn't have that prefix in its own name. As I've described, the *Sony* DRM software takes advantage of this behaviour by naming one if its services 'sysDRMServer'. The software consists of several other drivers that it names similarly, and the Registry key name for Aries itself is 'sysaries', so that it is not visible to applications like Regedit and security software when they enumerate the contents of HKLM\System\CurrentControlSet\Services.

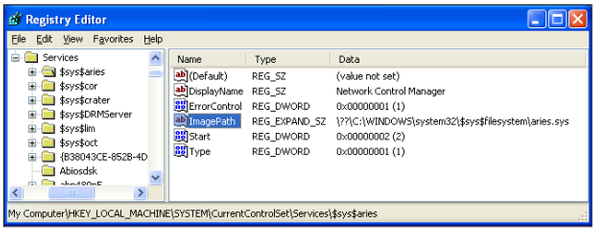


Figure 4: Aries Registry key.

Figure 4 shows the contents of the sysaries key after the Aries driver is disabled and the key is visible.

The Aries driver registers an unload function and takes other steps that suggest that the developer believed a hook-driver can be unloaded safely.

In fact, *Sony*'s decloaking patch unloads the driver while the system is running, which can lead to a crash. Each of the hook functions calls the Plug and Play Manager function IoAcquireRemoveLockEx on entry and IoReleaseRemoveLockEx on exit and the driver's unload function executes the two functions in sequence and then pauses for a short time before exiting. The names of the functions imply that they synchronize safe driver unloading with the I/O system, but in reality they simply increment and decrement a reference count. Standard Plug and Play drivers leverage the reference to unload only when their devices are not in use, but Aries is a non-Plug and Play driver so the counter plays no role in its unload logic.

There's no way for a driver that hooks system calls to guarantee that other threads in the system will not attempt to execute the hook functions after the driver unloads. A race condition exists where, for example, a thread is pre-empted just as it is about to execute the first few instructions of a hook function, the driver unloads, and then the thread executes invalid memory the driver just occupied. This scenario is unlikely, but the Aries unload logic demonstrates an ignorance of both *Windows* device driver and multi-threaded programming.

THE SECURITY AND RELIABILITY OF ARIES

An obvious security problem created as a side effect of the Aries cloak is that other applications can take advantage of its general nature to hide their own objects from end users.

Malware authors released several viruses shortly after I disclosed the existence of Aries that creates objects with the magic 'sys' prefix, and World of Warcraft (WoW) gamers published a way to circumvent the WoW anti-cheat system by hiding executable images behind the Aries cloak.

The media was quick to claim that Aries opens huge security holes, but the fact is that viruses could just as easily deploy their own rootkits instead of piggy-backing on Aries. The security implications of rootkits are complicated and are tied directly to the visibility of the software that they cloak.

In general, I believe that rootkits pose a security risk not because of potential errors in their cloaking, but because the software they cloak is generally completely invisible to systems administrators. Virtually all software has security flaws, but systems administrators can check periodically or use patch-management software to ensure that systems are updated with the latest fixes, or they can uninstall any software that they deem to be a risk. However, there is no advertisement to users of the presence of the *Sony* DRM software by way of auto-updater or bundled uninstall utility, so it can't be patched or uninstalled. As a result, a security problem in any of its components is permanent and undetectable.

The other concern about cloaked kernel-mode code is reliability. Bugs in the Aries driver could impact the stability of *Windows* and lead to system crashes.

My analysis shows that the Aries driver contains at least one bug that can lead to a crash. All but one of the hook functions filter data coming out of the kernel and so can trust the validity of the buffers on which they operate. The hook for ZwCreateFile, however, checks the file name input parameter for the 'sys' prefix and therefore accesses pointers that are passed by applications and potentially invalid. Aries omits validation of the input buffer, however, and so can easily be directed to perform an invalid memory access from kernel-mode, which is a violation that causes the Windows Memory Manager to crash the system. The NtCrash2 program that I wrote in 1997 to stress the input parameter validation of the *Windows* kernel triggers this bug, as can be seen in the bluescreen example shown in Figure 5.

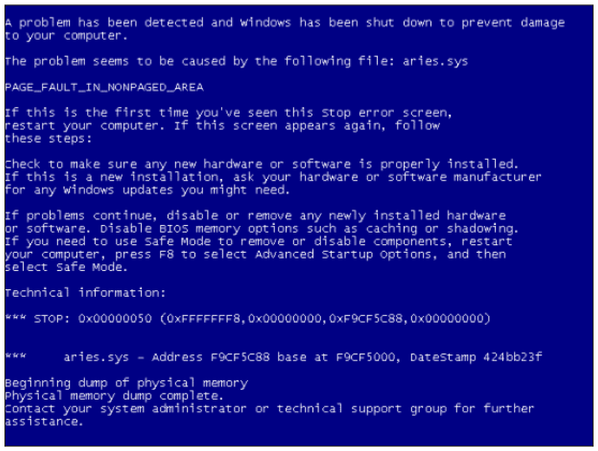


Figure 5: Aries blue screen crash.

The reliability risks caused by rootkits are similar to their security risks. While bugs in user-mode processes manifest as isolated crashes of those processes, bugs in drivers, including Aries and the other XCP drivers, result in *Windows* crashes. Crash analysis of resulting dump files might identify the problematic driver, but because the drivers and the Registry keys that configure them to load are cloaked while *Windows* is online, an administrator has no way of disabling or removing the drivers. If the sequence of execution that triggers a bug is unavoidable then the installation becomes unusable. The XCP software exacerbates this problem by configuring most of its drivers so that they also load in Safe Mode.

COMMERCIAL ROOTKITS

The furor over *Sony*'s rootkit centres on lack of disclosure during the installation process and the rootkit's use in concealing associated software from users. The resulting security and reliability risks only highlight the negative impacts of hidden software. However, are rootkits always bad?

Two other commercial products, both sold by security vendors, utilize rootkit technology: *Symantec*'s *Norton Undelete* and *Kaspersky Antivirus* (*KAV*). *Norton* uses a rootkit to hide the presence of directories in which it stores backup copies of files deleted by end users so that users cannot accidentally delete the backups. *KAV* stores a file's scanning information in an NTFS alternate data stream that it attaches to the file. It hides the streams so that they don't perturb the appearance of files when the files are read by stream-aware applications.

These examples differ greatly from the *Sony* case: *Symantec* and *Kaspersky* use rootkits in a way that is intended to benefit the consumer and not the software vendor. Also, the software that utilizes the rootkit advertises its presence to the user, implements auto-update features, and can easily be uninstalled. However, although the security and reliability risks of these rootkits are minimal, there is still potential for exploitation by the same type of opportunistic malware that uses the Aries cloak. In the end, I believe that the benefits of even benign cloaking are generally outweighed by the potential risks, and that software vendors should use alternative technologies if possible.

CALL FOR PAPERS

VB2006 MONTRÉAL

Virus Bulletin is seeking submissions from those wishing to present at VB2006, the Sixteenth Virus Bulletin International Conference, which will take place 11–13 October 2006 at the Fairmont The Queen Elizabeth, Montréal, Canada.

The conference will include three days of 40-minute presentations running in two (2) concurrent streams: Technical and Corporate. Submissions are invited on all subjects relevant to anti-malware and anti-spam. In particular, *VB* welcomes the submission of papers that will provide delegates with ideas, advice and/or practical techniques, and encourages presentations that include practical demonstrations of techniques or new technologies.

SUGGESTED TOPICS

The following is a list of topics suggested by the attendees of VB2005. Please note that this list is not exhaustive, and papers on these and any other anti-malware and spam-related subjects will be considered.

TECHNICAL AV

- Malware collection tools/honeypots
- Malware classification tools
- Spyware/adware – definition, techniques and detection
- Malware with respect to cryptography
- Threats and protection for mobile devices
- Emulation, engine level sandboxing, unpacking techniques
- Rootkits
- x64 malware
- Malware on non-*Windows* platforms
- Emulators/heuristics/PE unpacking on non-*Windows* platforms
- Hardware anti-malware solutions
- Tools of the trade (deobfuscation, IR, etc.)
- Behavioural analysis and detection
- Proof of concept demonstrations
- Phishing
- Latest malware outbreaks
- Wireless security
- CERT/vendor cooperation
- Hardware supported virtualisation

CORPORATE AV

- Case studies
- Best practices
- Policy enforcement
- Vulnerability assessment
- Risk management
- Phishing & fraud in the corporate environment
- Spyware/adware in the corporate environment
- Online crime prevention
- Tracing malware authors/perpetrators
- Anti-virus & anti-spyware performance testing
- Educating users
- Management of anti-virus infrastructures
- Proactive detection mechanisms
- IDS/IPS
- False positive prevention
- Government security policies
- IT outsourcing and associated risks
- Mobile threats
- Anti-malware managed services

SPAM

- Experience with spam filters in the corporate environment
- Best practices
- Spam and spammers from a legal point of view
- Spam filter performance testing
- Latest spam filter avoidance techniques (spammers' tricks)
- Latest anti-spam techniques
- Filtering phishing mails

HOW TO SUBMIT A PAPER

Abstracts of approximately 200 words must be sent as plain text files to editor@virusbtn.com no later than **Thursday 9 March 2006**. Submissions received after this date will not be considered. Please include full contact details with each submission.

Following the close of the call for papers all submissions will be anonymised before being reviewed by a selection committee; authors will be notified of the status of their paper by email. Authors are advised that, should their paper be selected for the conference programme, the deadline for submission of the completed papers will be Monday 5 June 2006 and that full papers should not exceed 6,000 words. Further details of the paper submission and selection process are available at http://www.virusbtn.com/conference/.

COMPARATIVE REVIEW

WINDOWS SERVER 2003 ENTERPRISE X64 VERSION

Matt Ham

Over the last year I have received an increasing number of enquiries, from both product developers and end users, as to when *Virus Bulletin* would produce a review on a 64-bit operating system. This comparative review comes as a result of that interest.

With 64-bit systems there is a range of hardware available, with operating systems to match. Having asked a selection of vendors and end users, it seems that *Athlon 64* processors are the most commonly used with 64-bit operating systems and thus were chosen as a hardware platform. *Windows Server 2003 x64* version was selected as the operating system, again based on reports received from a number of vendors and end users.

The biggest surprise in the review was the lack of submissions. I was certainly expecting a smaller number of products to be submitted for this test than for the previous *Windows 2003 Server* review, but for numbers to drop to just over a third was more extreme than expected. Whether other products were missing due to corporate cowardice or known incompatibilities with the platform I will leave to the reader to imagine.

TEST SETS

With hardware and operating systems already changed drastically it seemed unwise to make major changes to the test sets too. In the event, the most recent WildList available at the start of the test period was that from July 2005 – only one month newer than the one used in October's *Windows Server 2003* comparative review (see *VB*, October 2005, p.12). All the products included in this review were also tested on that occasion. Products were dated no later than 31 October 2005.

That is not to say that there wasn't a great temptation to add new samples to both clean and infected test sets on this occasion. The clean test sets in particular are perhaps unrepresentatively high in dynamic archives, which slow on-demand scan speeds more than would be seen in most real-world settings. Both test sets will be updated considerably between now and mid-2006 – and had there not already been so many other major changes this month, the process would already have begun.

Since this was the first outing of this hardware, the throughput tests cannot be compared directly with past results. In future reviews the hardware is likely to vary between tests, so care should be taken to ensure than any comparison is meaningful.

Some clarification has been requested as to the way in which our tests are performed where archives are concerned. In all cases the non-archive clean test sets are scanned using the product's default settings. In some cases, however, the product's default settings do not include the scanning of ZIP archives. For these products archive scanning is activated during the archive throughput tests, but not at any other time. This avoids creating the illusion of those products with no default archive scanning having astoundingly speedy throughput on archives.

Archive scanning becomes more of a thorny issue where dynamic archives are concerned. A product which does not scan such files will have a distinct advantage in scanning the clean test set over a product where such a setting is off by default. This is a genuine real-world difference, although, as mentioned above, the throughput results are somewhat biased towards products with either very fast or non-existent handling of dynamically compressed executables.

Alwil avast! 4.6.511

ItW Overall	100.00%	**Macro**	99.56%
ItW Overall (o/a)	100.00%	**Standard**	99.36%
ItW File	100.00%	**Polymorphic**	93.57%

Avast!'s on-access scanner is still one of the more fussy with respect to what will cause a detection to be announced. As a result, detections were logged on access when copying files, rather than simply accessing the files. *Avast!* is also one of the products where ZIP scanning was activated for the purposes of archive throughput testing. *Avast!* began a fairly predictable trend in which products behaved almost exactly as they did in the previous *Windows 2003 Server* review. A VB 100% award was the result again.

CA eTrust Antivirus 7.1.192 (InoculateIT engine) 23.70 86

ItW Overall	100.00%	**Macro**	99.90%
ItW Overall (o/a)	100.00%	**Standard**	99.63%
ItW File	100.00%	**Polymorphic**	99.89%

Although supplied as part of the standard *eTrust* package, the *InoculateIT* engine is not the default for this product, and as a result does not qualify for a VB 100% award – the results are presented here purely for interest. True to recent form, the product put in a very good performance, with all infected files In the Wild detected as such.

On-access tests	ItW File		ItW Boot		ItW Overall	Macro		Polymorphic		Standard	
	Number missed	%	Number missed	%	%	Number missed	%	Number missed	%	Number missed	%
Alwil avast!	0	100.00%	0	100.00%	100.00%	18	99.56%	112	93.58%	17	99.18%
CA eTrust Antivirus (I)	0	100.00%	0	100.00%	100.00%	4	99.90%	1	99.89%	4	99.51%
CA eTrust Antivirus (V)	0	100.00%	0	100.00%	100.00%	12	99.82%	1	99.95%	3	99.84%
CAT Quick Heal	0	100.00%	0	100.00%	100.00%	82	98.04%	313	96.25%	147	93.06%
Eset NOD32	0	100.00%	0	100.00%	100.00%	0	100.00%	0	100.00%	0	100.00%
GDATA AntiVirusKit	0	100.00%	3	0.00%	99.55%	0	100.00%	0	100.00%	0	100.00%
Grisoft AVG	0	100.00%	0	100.00%	100.00%	0	100.00%	257	85.97%	30	98.41%
Kaspersky KAV	0	100.00%	0	100.00%	100.00%	0	100.00%	0	100.00%	2	99.88%
McAfee VirusScan	0	100.00%	0	100.00%	100.00%	0	100.00%	29	97.67%	0	100.00%
Symantec SAV	0	100.00%	0	100.00%	100.00%	0	100.00%	0	100.00%	0	100.00%

CA eTrust Antivirus 7.1.192 (Vet engine) 11.9 9487

ItW Overall	100.00%	**Macro**	99.82%
ItW Overall (o/a)	100.00%	**Standard**	99.96%
ItW File	100.00%	**Polymorphic**	99.95%

While the log files for both the incarnations of *eTrust* continue to be the epitome of uselessness, the product itself performs well in both throughput and detection tests. A VB 100% award is the result, even though it would not be readily apparent from normal scrutiny of the aforementioned logs.

CAT Quick Heal 8.00 SP1

ItW Overall	100.00%	**Macro**	98.18%
ItW Overall (o/a)	100.00%	**Standard**	96.48%
ItW File	100.00%	**Polymorphic**	96.25%

This submission was designated the server version of the product, which in many other cases tends to result in a rather complex installation procedure.

Quick Heal proved to be at quite the opposite end of the spectrum, with perhaps the fastest install procedure of any GUI-based anti-virus program I have reviewed. Scanning too was relatively rapid and resulted in detection of all the samples in the In the Wild (ItW) test sets – both on demand and on access. It comes as no surprise that this performance is rewarded with a VB 100%.

ESET NOD32 1.1268(20051031)

ItW Overall	100.00%	**Macro**	100.00%
ItW Overall (o/a)	100.00%	**Standard**	100.00%
ItW File	100.00%	**Polymorphic**	100.00%

Having been tested in a comparative review two months ago and a standalone review in last month's issue of the magazine (see *VB*, November 2005, p.16), no great surprises were expected from *NOD32*. *NOD32* has ZIP archive scanning turned off by default, although W32/Heidi.A is detected by the engine as a special case, accounting for full detection of this virus in the standard test set. In any case detection was at its usual high levels for this product, and *NOD32* obtains a VB 100% award for its collection as a result.

GDATA AntiVirusKit 16.0.3

ItW Overall	100.00%	**Macro**	100.00%
ItW Overall (o/a)	99.55%	**Standard**	100.00%
ItW File	100.00%	**Polymorphic**	100.00%

GDATA's product is one which has suffered a little in the past from sluggish scanning – a result of the fact that it has two scanning engines which are both in use in each scan. However, the additional raw power of the new hardware in

use here made this less noticeable during testing. The file-based part of the testing was a definite success for *AVK*, with all infected files detected both on access and on demand. However, scanning of floppies on access proved less of a triumph. No detection could be triggered in any log, and no alerts were generated for infected disks. Whether this was due to the change in platform or hardware will no doubt be a point of investigation for the *GDATA* developers. *AVK* thus fails to obtain a VB 100% award on this occasion.

Grisoft AVG Anti-Virus 7.1.362

ItW Overall	100.00%	**Macro**	100.00%
ItW Overall (o/a)	100.00%	**Standard**	98.56%
ItW File	100.00%	**Polymorphic**	85.97%

The installation files for *AVG* are the same across all recent *Windows* platforms, with no reboot required before installation is declared complete. As usual, however, the machine was rebooted after installation as part of the standard test regime. The scanner detected all files in the ItW test set as in previous tests. With no false positives, *AVG* is worthy of a VB 100%. Indeed the whole test was notable for the fact that no false positives were generated.

Kaspersky Anti-Virus 5.0.70.0

ItW Overall	100.00%	**Macro**	100.00%
ItW Overall (o/a)	100.00%	**Standard**	100.00%
ItW File	100.00%	**Polymorphic**	100.00%

The *Kaspersky* product tested here was the command line scanner, the optional GUI being left for examination in future. The on-demand scanner still seemed to be a little slower than expected when scanning infected files. However, this was only in comparison with *Kaspersky*'s usual rapid throughput, its speed of scanning coming nowhere close to slow overall.

With one hundred per cent detection on demand, and very close to this on access (including full detection of all ItW samples), a VB 100% award is on its way to Moscow.

McAfee VirusScan Enterprise 8.0.0 4616 4400

ItW Overall	100.00%	**Macro**	100.00%
ItW Overall (o/a)	100.00%	**Standard**	100.00%
ItW File	100.00%	**Polymorphic**	100.00%

The biggest niggle with *VirusScan* is the highly involved process that is required to set up new scans, however this is gradually showing signs of improvement. Oddly, on this occasion, changing and saving the settings resulted in a second prompt to save when the task was closed.

Scanning completed with no problems at all and detection rates were very good as expected – all files being detected in the on-demand scan of infected objects. Files missed on access included several samples of W32/Etap, presumably due to the complexity of these files causing a time-out

On-demand tests	ItW File		ItW Boot		ItW Overall	Macro		Polymorphic		Standard	
	Number missed	%	Number missed	%	%	Number missed	%	Number missed	%	Number missed	%
Alwil avast!	0	100.00%	0	100.00%	100.00%	18	99.56%	113	93.57%	15	99.36%
CA eTrust Antivirus (I)	0	100.00%	0	100.00%	100.00%	4	99.90%	1	99.89%	2	99.63%
CA eTrust Antivirus (V)	0	100.00%	0	100.00%	100.00%	12	99.82%	1	99.95%	1	99.96%
CAT Quick Heal	0	100.00%	0	100.00%	100.00%	75	98.18%	313	96.25%	100	96.48%
Eset NOD32	0	100.00%	0	100.00%	100.00%	0	100.00%	0	100.00%	0	100.00%
GDATA AntiVirusKit	0	100.00%	0	100.00%	100.00%	0	100.00%	0	100.00%	0	100.00%
Grisoft AVG	0	100.00%	0	100.00%	100.00%	0	100.00%	257	85.97%	27	98.56%
Kaspersky KAV	0	100.00%	0	100.00%	100.00%	0	100.00%	0	100.00%	0	100.00%
McAfee VirusScan	0	100.00%	0	100.00%	100.00%	0	100.00%	0	100.00%	0	100.00%
Symantec SAV	0	100.00%	0	100.00%	100.00%	0	100.00%	0	100.00%	0	100.00%

Hard Disk Scan Rate	Executables			OLE Files			Zipped Executables		Zipped OLE Files	
	Time (s)	Throughput (kB/s)	FPs [susp]	Time(s)	Throughput (kB/s)	FPs [susp]	Time (s)	Throughput (kB/s)	Time(s)	Throughput (kB/s)
Alwil avast!	56.0	9766.6	0	13	6102.6	0	45	3542.6	14	5329.1
CA eTrust Antivirus (I)	53.0	10319.5	0	2	39666.9	0	21	7591.3	4	18651.9
CA eTrust Antivirus (V)	44.0	12430.3	0	1	79333.8	0	23	6931.2	4	18651.9
CAT Quick Heal	38.0	14393.0	0	6	13222.3	0	22	7246.2	6	12434.6
Eset NOD32	18.0	30385.1	0	2	39666.9	0	13	12262.8	3	24869.2
GDATA AntiVirusKit	194.0	2819.2	0	7	11333.4	0	74	2154.3	10	7460.7
Grisoft AVG	75.0	7292.4	0	3	26444.6	0	28	5693.4	4	18651.9
Kaspersky KAV	85.0	6434.5	0	8	9916.7	0	40	3985.4	9	8289.7
McAfee VirusScan	42.0	13022.2	0	5	15866.8	0	27	5904.3	6	12434.6
Symantec SAV	79.0	6923.2	0	7	11333.4	0	33	4830.8	5	14921.5

somewhere in scanning. *VirusScan* also required the scanning of ZIP files to be activated for the archive clean set tests. With full detection of all samples in the ItW test sets, however, a VB 100 % award is the result.

Symantec AntiVirus 10.0.0.359 103.0.2.7

ItW Overall	100.00%	**Macro**	100.00%
ItW Overall (o/a)	100.00%	**Standard**	100.00%
ItW File	100.00%	**Polymorphic**	100.00%

Things got off to a bad start with *SAV*, the initial package supplied being for the *Itanium* processor rather than the *AMD* used in the tests. *Symantec* has discontinued support for *Itanium* processors in *SAV 10*, so this should not be a problem in future.

More of an issue, however, was the speed at which infected files were scanned on demand. At around four seconds per file, this was over 1,000 times slower than some of the other products on test. In fact, even the total scan times of all other tests performed during the course of the review did not reach that of *SAV*'s single on-demand scan. The problem seemed to be linked in some way to the GUI, since on-access scanning proceeded at a far more reasonable speed. What is more, after a large number of infected files had been scanned on demand, the load and unload times of the GUI rose to close to five minutes each.

The *SAV* log file continues to seem to be the product of a madman or a fool – for example, several samples of W97M/AntiSocial.F were logged under the highly useful file name of '??????'. Needless to say, this is not a name which any of the samples possess, the real name of this file being ANTI_F-1.DOC.

However, *SAV* did manage to detect all the samples in the ItW test sets, and despite the fact that I have had more pleasurable dentistry, a VB 100% is awarded to this product.

CONCLUSIONS

In the aftermath of the tests it has become clear that the tales of woe I had heard concerning 64-bit operating systems were, at least as far as anti-virus software is concerned, somewhat exaggerated. Installation of the operating system and drivers proceeded without a hitch and the same was true for the majority of the products on test. In many cases the product submitted was exactly the same as that supplied for the previous test, the installation packages combining 64-bit and 32-bit versions of the application.

After such a painless review I expect the next 64-bit comparative review in these pages to be graced with a rather larger number of entrants.

Technical details

Test environment: Identical AMD Athlon 64 3800+ dual core machines with 1GB RAM, 40GB and 200 GB dual hard disks, DVD/CD-ROM and 3.5-inch floppy drive, running *Microsoft Windows Server 2003 Enterprise X64 version*, *Service Pack 1*.

Virus test sets: Complete listings of the test sets used are at http://www.virusbtn.com/Comparatives/Win64/2005/test_sets.html.

A complete description of the results calculation protocol is at http://www.virusbtn.com/Comparatives/Win95/199801/protocol.html.

END NOTES & NEWS

ACSAC 21 (the Applied Computer Security Associates' Annual Computer Security Conference) takes place 5–9 December 2005 in Tuscon, AZ, USA. The complete programme and online registration are available at http://www.acsac.org/.

Infosecurity USA will be held 6–8 December 2005 in New York, NY, USA. The conference will take place 6–8 December, with the accompanying exhibition running from 7–8 December. For details see http://www.infosecurityevent.com/.

The inaugural AVIEN/AVIEWS conference will take place from 11am to 4pm Eastern Standard Time on 18 January 2006 by webcast. Details of how to register will be released and circulated on the AVIEN and AVIEWS forums in due course – in the meantime the programme can be viewed at http://www.avien.org/.

The Black Hat Federal Briefings & Training takes place 23–25 January 2006 in Washington, DC, USA. Registration for the event is now open. The deadline for submission of papers is 15 December 2005. See http://www.blackhat.com/.

The Second Annual IFIP WG 11.9 International Conference on Digital Forensics will take place 29 January to 1 February 2006 in Orlando, FL, USA. The conference provides a forum for presenting original, unpublished research results and innovative ideas related to the extraction, analysis and preservation of all forms of electronic evidence. The conference is limited to 50 participants to facilitate interactions between researchers and intense discussions of critical research issues. Technical papers on all areas related to the theory and practice of digital forensics will be presented. For more details see http://www.cis.utulsa.edu/ifip119/.

IT-DEFENSE 2006 takes place 30 January to 3 February 2006 in Dresden, Germany. The event is aimed at network administrators, developers, DP managers, IT security officers, auditors, consultants and hackers seeking to exchange information and make contact with leaders in the industry. For more information see http://www.it-defense.de/.

RSA Conference 2006 will be held 13–17 February 2006 in San Jose, CA, USA. An early bird reduced registration rate is available for those who register before 18 November 2005. For more details see http://2006.rsaconference.com/us/.

The Black Hat Europe 2006 Briefings & Training will be held 28 February to 3 March 2006 in Amsterdam, The Netherlands. For details including online registration see http://www.blackhat.com/.

Infosecurity Europe 2006 takes place 25–27 April 2006 in London, UK. For details or to register interest in the event, see http://www.infosec.co.uk/.

The 15th EICAR conference will take place from 29 April to 2 May 2006 in Hamburg, Germany. Authors are invited to submit full papers and posters for the conference. The deadlines for submissions are as follows: academic papers (in full) 13 January 2006; poster presentations 24 February 2006. For more information, including the full call for papers, see http://conference.eicar.org/2006/.

The Seventh National Information Security Conference (NISC 7) will take place from 17–19 May 2006 at St. Andrews Bay Golf Resort & Spa, Scotland. Enquiries may be directed to tina.deighton@sapphire.net or via http://www.nisc.org.uk/ .

The Fourth International Workshop on Security in Information Systems, WOSIS-2006, will be held 23–24 May 2006 in Paphos, Cyprus. For details see http://www.iceis.org/.

CSI NetSec '06 takes place 12–14 June 2006 in Scottsdale, AZ, USA. Topics to be covered at the event include: wireless, remote access, attacks and countermeasures, intrusion prevention, forensics and current trends. For more details see http://www.gocsi.com/.

Black Hat USA 2006 will be held 29 July to 3 August 2006 in Las Vegas, NV, USA. Online registration will be available from 15 March. See http://www.blackhat.com/.

The 16th Virus Bulletin International Conference, VB2006, will take place 11–13 October 2006 in Montréal, Québec, Canada. See p.15 for the full call for papers. Online registration and further details will be available soon at http://www.virusbtn.com/.

SUBSCRIPTION RATES

Subscription price for 1 year (12 issues) including first-class/airmail delivery: £195 (US$358)

Editorial enquiries, subscription enquiries, orders and payments:

Virus Bulletin Ltd, The Pentagon, Abingdon Science Park, Abingdon, Oxfordshire OX14 3YP, England

Tel: +44 (0)1235 555139 Fax: +44 (0)1235 531889
Email: editorial@virusbtn.com Web: http://www.virusbtn.com/

vb Spam supplement

CONTENTS

S1 **NEWS & EVENTS**

S2 **FEATURE**

The spectre of parasitic spam

NEWS & EVENTS

RADIO LISTENERS GET THE VIAGRA MESSAGE

Dutch radio listeners are being warned of the dangers of buying Viagra from spammers. *Pfizer*, the manufacturer of the erectile dysfunction drug (and its patent holder), has launched a radio campaign to alert consumers to the fact that 97 per cent of the drug sold via spam is counterfeit. The announcements, which are being run both on public broadcast radio and on *Sky Radio*, warn that drugs that are advertised through spam are unlikely to have passed the stringent drug approval process and may not even contain the relevant active ingredients.

This is not the first time the pharmaceutical company has taken action to protect its customers (and its brand) from dubious drug peddling by spammers. Earlier this year *Pfizer* filed civil actions against websites that it accused of promoting and selling drugs that have not been approved by regulators, while *Microsoft* filed civil actions against the spammers advertising for those websites.

SIX YEARS FOR BRITISH SPAMMER

A young Briton has been jailed for six years after amassing £1.6 million through spamming and online scamming.

The 23-year-old 'weaselboy', whose real name is Peter Francis-Macrae, made most of his money through a fraudulent domain name registration 'service' for non-existent .eu domain names. He advertised the 'service' by sending thousands of unsolicited emails, then charged customers to pre-register the .eu domain names before their release by the regulatory body. A police spokesman said at the end of the investigation that they had received more than 2,000 complaints about Francis-Macrae's online activities from complainants around the world.

The court found Francis-Macrae guilty of a catalogue of offences, including fraudulent trading, concealing criminal property, threatening to destroy or damage property, making threats to kill, and blackmail. He was sentenced to a total of six years for the combination of offences.

PASSPORT TO THE COURT ROOM

Mobile network provider *Verizon* has filed a lawsuit against a travel company which it alleges sent unsolicited SMS messages to customers on its network.

According to *Verizon*, *Passport Holidays* sent over 98,000 unsolicited SMS messages to its customers. The message read 'you just WON a Cruise to the Bahamas' and instructed the recipient to call an 800 number to claim their prize.

The travel company told *Verizon* that the owner of every phone number it had contacted had opted in to receive the messages. However, *Verizon* contends that its customers had not given consent to be contacted. Furthermore, the fact that many of the messages were sent to sequential numbers within a short space of time indicates that an automated dialler had been used – which is a violation of US regulations.

Verizon is seeking damages for every violation, as well as punitive damages.

SPAMMERS SETTLE CHARGES

A group of spammers have paid $621,000 in settlement of charges brought against them by the US Federal Trade Commission (FTC).

The complaint, which was filed in January, charged that the five individuals and their associated companies were using spam both directly and through affiliates to advertise and sell access to sexually explicit content online.

The emails were found to violate the CAN-SPAM Act and the FTC's Adult Labelling Rule on several counts: by failing to include a label for sexually explicit content; displaying sexually explicit content in the email itself; using misleading header information; using misleading subject lines; failing to include an opt-out notice; failing to have a functioning opt-out mechanism; failing to identify the

emails as advertisements and failing to provide a valid physical postal address.

The settlement order demands that the defendants clean up their act, stipulating that they must include an opt-out mechanism in any future marketing emails they send, label any sexually explicit email and keep sexually explicit content out of the subject line and the initially viewable area of the emails. The order also requires that the defendants monitor their affiliates closely to ensure that they also comply with the regulations.

USERS GET MORE HELP TO AVOID PHISHING ATTACKS

Microsoft has upped its effort to help customers avoid phishing attacks by signing agreements with three data providers, *Cyota*, *Internet Identity* and *MarkMonitor*, which will provide the company with up-to-date information on known phishing websites.

The Microsoft Phishing Filter (to be incorporated into *IE*, *Hotmail* and *Windows Live*) will use a 'URL reputation service', which will receive constant updates about the legitimacy of websites across the Internet. The filter will also scan any web page the user visits for traits associated with phishing scams.

Microsoft is also in talks with other web browser developers, including *Mozilla* and *Opera*, in an attempt to develop a standard means of providing information about the trustworthiness of a website. While the current yellow padlock symbol – signifying that a site is encrypted – is more or less universally recognised by users, it is not a guarantee of the site's trustworthiness and certification authorities are seeking a more rigorous way of creating new 'high assurance' digital certificates. Although individual browser developers have introduced their own methods of identifying untrustworthy sites to their users – such as turning the address bar red when the user visits a known phishing site (in *Microsoft*'s *IE 7*) – they hope to come to an agreement on a standard way of presenting the information.

EVENTS

The third Conference on Email and Anti-Spam, CEAS 2006, will be held 27–28 July 2006 in Mountain View, CA, USA. The conference encompasses a broad range of issues relating to email and Internet communication. The conference format includes short and long presentations selected by peer review, as well as invited addresses. Those wishing to present long or short papers are invited to submit their proposals before 23 March 2006. Full details can be found at http://www.ceas.cc/.

FEATURE

THE SPECTRE OF PARASITIC SPAM

Dagmar Swimmer
Independent consultant, Switzerland

With the increasing use of bot networks by spammers, Nick FitzGerald pointed out recently [1] that all bets were off with respect to what the spammer might do and therefore how effective anti-spam measures like authentication might be. In particular, one type of spam that we haven't seen before might now be on the horizon.

Until now, bots have given spammers the means of sending spam decentrally, which both diffuses the spammers' identity and increases their output through distributed processing. But the same bots could also be used for something more subtle: modifying existing and legitimate email traffic for the purpose of spamming. This is similar to adding a flyer to outgoing envelopes, piggybacking an advertisement on top of other correspondence. It is also similar to droppers that insert malware into existing benign programs.

In the following article we will explore this new type of spam, which I call 'parasitic spam' (or p-spam for short), and look at the consequences.

SPAM ON HAM, PLEASE!

In a nutshell, parasitic spam is spam that is attached to legitimate email, or ham. If the reader is reminded of something in this concept, then it is probably the dubious custom that many free email services practise of appending an advertisement for their services (or others') to each email the user sends. Just like spam, these appended messages are unsolicited, commercial, do not originate with the sender and marginally increase the bulk of the email traffic.

For example:

```
From: Joe
To: Jane

message text

Cheers, Joe
--
Get your free email account at http://free_email.nu
```

Typically, the free email service providers do not offer a method of opting out, from either the sender or the receiver side. Because these 'signatures' are perceived to be a nuisance, some mailing list software includes tools that will strip them, but this only works because they tend to be relatively predictable.

P-spam is similar but more devilish. First let's assume that there exist bots with actual users on the machine or that the bot in some way controls some email traffic – perhaps because it sits on a machine that is a mail relay. The spam bot watches for outgoing or relayed email messages and modifies each before allowing it to be delivered. The result is that ham messages from legitimate senders to legitimate recipients now also include spam.

From the point of view of a context-unaware spam classifier, the message looks like spam with some ham-like scraped content inserted, and the filter has no way of knowing that it is essentially ham and should not be blocked. This is the crucial point of this attack: the p-spam cannot be blocked because that would cause a false positive as the message contains legitimate content, but to the spam classifier if looks like spam with scraped content.

THE SHAPE OF P-SPAM

So far, and to the best of my knowledge, p-spam has not been observed in the wild.

However, we can speculate on how such spam would look if it existed. One straightforward method is to add the spam to the ham by placing both as two parts of a multipart/mixed message. Schematically, that looks like this:

```
Content-Type: multipart/mixed; boundary="break"

--break
Content-Type: text/plain

ham
--break
Content-Type: text/plain

spam
--break
```

Standard MUA (Mail User Agent) rendering will display both parts in order of occurrence. This would be relatively easy to implement and reasonably unobtrusive because the ham is still visible and the spam part merely appended.

However, the MIME structuring also makes it easier to distinguish the spam and ham parts if we apply the spam classifier to each message part separately. A more malicious and blatant approach is to use multipart/alternative, for example:

```
Content-Type: multipart/alternative; boundary="break"

--break
Content-Type: text/plain

ham
--break
Content-Type: text/plain

spam
--break
```

Most MUAs will display just the spam part and not the ham if they are compliant with RFC1341 [2], and therefore this is more disruptive. The user is likely to vote it spam as they will not see the ham, and they run the risk of never seeing email from that sender again. The other content types, multipart/digest and multipart/parallel, can be misused in similar ways.

Another class of attack involves modifying the message but not the MIME structure. We have already seen the mechanics of appending an advertisement 'signature' to a message. Although not covered by an RFC, most MUAs recognize a '-- ' line to be the start of the signature and so these are easily discernible from the main message. If the perpetrator is acting by this convention, most spam classifiers should be able to adapt.

Alternatively, the spam could merely be appended to the ham without a separator or mixed into the text. An attack which would perhaps be more devious against HTML email would be to replace only the HREF part of the <A> tag or to insert an active script of some sort. Surely, there are endless variations of this form of attack. These cases will be hard for spam classifiers to deal with correctly, especially if they are also expected to remove the spam part.

EAT YOUR P-SPAM!

Is p-spam a good thing for spammers? Well, on the downside, it is a low-volume affair compared to the deluge we experience in common unsolicited bulk email. Furthermore, it also requires code to intercept and modify email via the TCP stack, APIs, in message queues, or however. The latter issue can be solved relatively easily by a motivated programmer and the former may not be perceived to be a problem if there is a much better return rate than current methods.

All the email will reach a legitimate target and sender authentication schemes will not work to prevent this sort of spam. (They will, of course, work as specified and probably help the email get through the spam filters.) P-spam does rely on an extensive bot network, but given the current state of security, it is probably not a problem for the hackers-for-hire to ensure this for a while yet.

The positive aspects for spammers, therefore, may outweigh the downsides. This type of spam cannot be blocked without causing a false positive, the recipient and sender are both genuine, and clever design may bring a high rate of return.

SO, WHAT DO WE DO?

A solution seems to require more context information than is contained in the message itself. One way of

providing context is by profiling normal email usage. Hershkop *et al*. [3] have written about an interesting approach to behaviour-based spam and virus detection, which could be adapted for p-spam detection. The paradigms should be familiar to those who have followed Intrusion Detection Systems research as a much discussed but rarely applied methodology that started the field back in the 1980s.

The premise is simple: profile the normal email use of known email users during a training phase and match against this profile during the later detection phase. By their nature, such systems must know about the email accounts in use in the system but could also be applied to incoming email. Knowledge of the users makes this approach different to common anti-spam filters that operate only on the content of the email with no regard for the context of the email.

The mechanics are based on statistics. A set of measurements are compiled within a certain time window. During the next time window, the same measurements are made and a distance function that compares the benchmark to the current measurements is used to determine how abnormal the behaviour was. If the behaviour was not significantly different, the measurements are averaged and this becomes the new benchmark. The distance function and the threshold of 'abnormality' must be chosen very carefully in order to avoid false positives and false negatives. Also, the types of measurement made must be relevant to the intended goals and ideally statistically independent of each other.

For the task of detecting p-spam, we might be looking for measurements like number of multipart/alternative parts that are of the same content type. Other measurements might be the size of the signature after the '-- ' divider. It may be normal for the user to append a large signature, but abnormal for him/her to create multiple parts of the same type, and these should show up in the statistics as abnormalities.

The reason that this technology is not being applied in the intrusion detection field is the false positive rate and the maintenance overhead. It is likely that the same problems will beset its application to p-spam unless the system is well thought through.

Furthermore, the methodology enables us to detect p-spam, but does not help us remove the offending spam section. A useful role of behaviour-based p-spam detection would be in combination with a spam classifier for email parts, but no one so far has shown how this could be done.

Existing anti-spam systems might be able to detect the spam parts of the p-spam using sliding windows or other methods that do not take the entire message into consideration. However, most of the current breed of anti-spam products are not set up to perform major message rewriting. Even if existing email classification methods can be tweaked to detect p-spam, if the users call for the spam to be removed but want the ham delivered, many systems simply won't be able to support this in the way in which they are currently designed.

P-SPAM PAYDAY

Given the way current spam classifiers operate, there is a small chance that p-spam already exists and is just being filtered out. This is a worrying scenario because it has direct economical consequences, but currently it seems unlikely that this is the case. It also shows that the use of p-spam might very well be a lose-lose situation, whereby the user loses delivery of his email, but the spammers do not get the return on investment this method promises either. The idea may therefore never get off the ground and that would be good to a degree.

It is also hoped that spammers become discouraged by the relatively low output volume versus the administrative costs of maintaining a bot network. There are many other spam methods that they could try, and these may seem more promising to them.

However, despite the good chance that the attack will not be tried any time soon, I would still encourage email operators and users to monitor the output of their spam filters for p-spam and for the vendors of spam filters to take the proactive step of reviewing their architecture to accommodate message rewriting so that they are not taken by surprise if the spammers find a way of making p-spam pay.

REFERENCES

[1] FitzGerald, Nick; 'Why "user authentication" is a bad idea', in *Proceedings of the Virus Bulletin Conference*, October 2005 (only abstract).

[2] Borenstein, N. and Freed, N.; 'MIME (Multipurpose Internet Mail Extensions)', IETF, RFC 1341, June 1992.

[3] Stolfo, Salvatore J.; Hershkop, Shlomo; Wang, Ke; Nimerkern, Olivier and Hu, Chia-Wei; 'A Behavior-based Approach to Securing Email Systems', *Mathematical Methods, Models and Architectures for Computer Networks Security*, Proceedings published by Springer Verlag, September 2003.