AUGUST 2006

virus BULLETIN

The International Publication on Computer Virus Prevention, Recognition and Removal

CONTENTS

IN THIS ISSUE

A FIRST FOR WEBMAIL...

In the past, we have seen email worms that sent themselves via *Outlook* and those that communicated with SMTP servers directly, but we hadn't seen an email worm that actually harnessed a webmail interface – until now. Eric Chien has all the details of JS.Yamanner@m, the first webmail worm.
page 4

...NOT A FIRST FOR STAROFFICE

A macro virus for *StarOffice* or merely an intended? Vesselin Bontchev sets the record straight on intended://StarBasic/Stardust.
page 7

VB 100% FOR NETWARE

John Hawes's first task as *VB*'s new Technical Consultant was to run a comparative review of AV products for *NetWare*. See how John and the eight products fared.
page 15

vbSpam supplement

This month: anti-spam news and events; and John Graham-Cumming puts forward his proposals for a unified naming scheme for spammers' and phishers' tricks.

ISSN 1749-7027

'You could be killed in either Bournemouth or Baghdad, but I know which destination I would be more concerned about.'

Graham Cluley, Sophos, UK

THE GREAT MAC DEBATE

I've just suffered a distributed denial-of-service attack. Not from a network of zombie computers under the control of an uber-hacker, but my inbox is creaking under the weight of the abusive email I have received from around the world.

The reason is that I dared to say something publicly that previously I've only said behind closed curtains, amongst trusted friends and family – something that has really, *really* annoyed some people: 'Have you thought about buying an Apple Mac instead?'

Yes, I hold my hands up. I dared to say the thing that a fair few in the security industry have appeared reluctant to say: there's an awful lot of malware for *Windows*, but hardly anything for Mac OS X.

I was spurred to say the unthinkable by some new research conducted by *SophosLabs*. An examination of the top malware seen at our global network of monitoring stations in the first half of 2006 found it was all *Windows*-specific. Not only that, but some of the biggest culprits (like the Netsky and Zafi worms) have been spreading successfully for a couple of years now. What's most interesting about these statistics, however, is what *doesn't* appear in the list. Apple Macintosh malware is nowhere to be seen. None of the malware in the chart can infect computers running Mac OS X.

It is still relatively rare for viruses to be written for Apple Macintosh computers. While the first malware for Mac OS X was seen in February 2006, it has not spread in the wild and has not spawned a flurry of other malicious code for Mac.

I like to think that businesses have woken up to the importance of running an up-to-date anti-virus product, and research suggests that most of them are recognising the importance of securing their systems properly. So it must be home users who are being infected by these old viruses. So, what are we going to do about home users like my Aunty Hilda who is never really going to get a grasp of computer security but still wants to email? The anti-virus industry has told users ad nauseum about the importance of running anti-virus, installing firewalls, applying patches and not opening unsolicited attachments. But worms, spyware and pornographic pop-ups are still hitting the average man in the street.

When I suggest to those home users that they might want to consider getting an Apple Mac next time, it is with good reason. My aim is to get them out of the hackers' firing line.

The issue here is analogous to advice the government might give people who are making travel plans. They might tell you that going to Iraq would put you at a greater risk of getting shot than going to the south coast of England, for instance. Yes, you could be killed in either Bournemouth or Baghdad, but I know which destination I would be more concerned about if my loved ones started packing their suitcases.

We've tried educating Joe Average about security for the last 20 years and he doesn't want to listen. He's not interested in hearing about the latest remove code execution vulnerability in the handling of WMF graphic files. But saying to users, 'You know, you'd be less prone to getting so many viruses if you used a Mac, because there are hardly any Mac viruses at all', is a message that many people would find easier to grasp.

Mac owners mustn't be complacent about security, of course, and should be sure to follow safe computing practices, but there can be no doubt that the vast majority of attacks are happening on *Windows*, leaving Mac users relatively unscathed. And that is something that home users may wish to consider if they're deliberating about the next computer they should purchase.

And my denial-of-service email attack? It hasn't come from angry *Microsoft Windows* users appalled that I'm suggesting some home users might benefit by switching to Mac. No, it has come from UNIX fans, accusing me of being part of a grand conspiracy not to promote their favourite OS instead. Sigh.

NEWS

SYSINTERNALS GOES THE MICROSOFT WAY

Microsoft announced the acquisition last month of privately held *Winternals Software LP* – the company responsible for the *Sysinternals* website and range of freeware tools. *Microsoft* has also 'acquired' the brains behind the company, Mark Russinovich and Bryce Cogswell – Mark will join the Microsoft Platforms & Services Division as a technical fellow, while Bryce will join the Windows Component Platform Team in the role of software architect.

The range of *Sysinternals* tools (including *Filemon*, *Regmon* and *RootkitRevealer*), are used extensively by systems administrators and security analysts across the world for systems troubleshooting, management and security. According to Mark Russinovich, the *Sysinternals* site will remain in its current state, the tools continuing to be free to download, while *Microsoft* determines the best way to integrate it into its own community efforts. Financial terms of the acquisition were not disclosed.

LINUX MAGAZINE PRINTS ROOTKIT HOW-TO

Imaginatively named magazine for *Linux* users *Linux Magazine* has published an article entitled 'How to write a rootkit'. The piece is the cover story for the August issue of the magazine.

According to the magazine the aim of the article is to arm systems administrators with the knowledge they need to stop rootkits – and anti-rootkit technology is examined elsewhere in the magazine. However, this does not detract from the fact that much of the cover article is devoted to an in-depth description of the routines required by a successful kernel rootkit – including example code.

While the writer suggests that rootkit techniques come in handy for the security-minded admin, the only example given is 'benign rootkit' Kernel Guard, which disables the kernel's module-loading functionality. The magazine goes on to analyse two *Linux* security systems – *AppArmor* and *SELinux*.

To see this piece of astonishingly irresponsible journalism for yourself, visit http://www.linux-magazine.com/issue/69/.

MORE ON THE XP COMPARATIVE

In *VB*'s June 2006 comparative review it was reported that the *Norman* product behaved badly, with repeated crashes on dealing with infected or previously disinfected files. *VB* would like to note that since then, neither *Norman*'s developers nor *VB*'s new resident product tester have been able to reproduce the bad behaviour described.

Prevalence Table – June 2006

Virus	Type	Incidents	Reports
Win32/Netsky	File	66,222	44.57%
Win32/Mytob	File	31,420	21.15%
Win32/Bagle	File	15,600	10.50%
Win32/Mydoom	File	15,429	10.38%
Win32/MyWife	File	9,281	6.25%
Win32/Lovgate	File	3,393	2.28%
Win32/Sdbot	File	2,251	1.51%
Win32/Pate	File	1,608	1.08%
Win32/Zafi	File	514	0.35%
Win32/Bugbear	File	462	0.31%
Win32/Funlove	File	460	0.31%
Win32/Feebs	File	417	0.28%
Win32/Bagz	File	353	0.24%
Win32/Sality	File	216	0.15%
Win32/Valla	File	147	0.10%
Win32/Mabutu	File	120	0.08%
Win32/Gibe	File	115	0.08%
Win32/Chir	File	112	0.08%
Win32/Brepibot	File	86	0.06%
Win32/Maslan	File	58	0.04%
Win32/Dumaru	File	53	0.04%
Win32/Mimail	File	36	0.02%
Win32/Scano	File	31	0.02%
Win32/Small	File	28	0.02%
Win32/Gael	File	27	0.02%
Win32/Klez	File	22	0.01%
Win32/Reatle	File	20	0.01%
Win32/Elkern	File	16	0.01%
Win32/Kedebe	File	13	0.01%
Win32/Magistr	File	12	0.01%
Wonka	Script	9	0.01%
Thus	Macro	6	0.00%
Others[1]		52	0.03%
Total		148,589	100%

[1]The Prevalence Table includes a total of 52 reports across 20 further viruses. Readers are reminded that a complete listing is posted at http://www.virusbtn.com/Prevalence/.

VIRUS ANALYSIS 1

MALICIOUS YAHOOLIGANS

Eric Chien
Symantec, Ireland

On 11 June 2006 we received reports of a worm propagating via email. That in itself was nothing special, but what was more interesting was the fact that this worm appeared to propagate only through *Yahoo! Mail* email accounts.

Once we obtained a sample, it became clear that this threat wasn't any ordinary email worm, but was the first webmail worm, later named JS.Yamanner@m.

In the past, we have seen email worms that sent themselves via *Outlook* and those that communicated with SMTP servers directly, but we hadn't seen an email worm that actually harnessed a webmail interface. JS.Yamanner@m utilized *Yahoo! Mail*'s webmail interface both to collect email addresses and to send itself to other users.

TARGET ELIMINATED

Furthermore, JS.Yamanner@m did not require a user to execute a file. Instead, the worm took advantage of a vulnerability in *Yahoo! Mail* so that merely by opening an infected mail message for reading, the user would cause the worm to execute and begin sending itself to addresses with which the infected user had corresponded in the past. This functionality would result in both the rise and the downfall of JS.Yamanner@m.

The vulnerability utilized by JS.Yamanner@m was an undisclosed zero-day vulnerability in *Yahoo!*'s JavaScript- and HTML-filtering algorithms.

Yahoo!, like other web applications, must parse emails for HTML and JavaScript and sanitize tag attributes that allow JavaScript execution. If *Yahoo!* didn't sanitize HTML and JavaScript contained in emails, then emails could contain malicious JavaScript that would be executed under the context of the *Yahoo!* domain. This could allow one to read other people's emails, impersonate other users, and create a self-replicating worm.

An example of a tag attribute that *Yahoo!* sanitizes is *onload*. The onload attribute instructs the browser to execute JavaScript code (or another scripting language) as soon as the page is rendered in the browser.

Another example is the target attribute. The target attribute instructs the browser to load the content in a particular page or frame. This page or frame could reside on another domain. If the tag is not filtered, then private content on a yahoo.com page, such as your email address, could be sent to a page outside of the yahoo.com domain.

In this case, the worm utilized the following malformed HTML:

```
<img src='http://us.i1.yimg.com/us.yimg.com/i/us/nt/
ma/ma_mail_1.gif' target=""onload="// malicious
javascript here //">
```

This HTML is nonsensical as the information within the quotes after the target attribute should contain the name of a frame or page, but instead includes nothing and is followed immediately by an onload attribute.

The purpose of this malformed HTML becomes clearer when one understands *Yahoo!*'s filtering algorithms. In particular, *Yahoo!* filters out the target attribute to prevent information disclosure. However, this results in the following HTML being rendered by the browser:

```
<img src='http://us.i1.yimg.com/us.yimg.com/i/us/nt/
ma/ma_mail_1.gif' onload="// malicious javascript
here //">
```

Note that the *target=""* text has been removed and this results in proper HTML. However, due to the fact that *Yahoo!* has removed the target attribute, the onload attribute now resides in a valid position and *Yahoo!*'s algorithms don't take a second pass at filtering. As a result, the dangerous onload attribute is not filtered out.

This vulnerability allows the JavaScript within the onload attribute to be executed by the browser under the context of the yahoo.com domain, all without any interaction from the user. The user must merely view the page.

The vulnerability has since been fixed by *Yahoo!* and now results in the following HTML:

```
<img src='http://us.i1.yimg.com/us.yimg.com/i/us/nt/
ma/ma_mail_1.gif' onfiltered="// malicious javascript
here //">
```

Note that, while the *target=""* text has still been removed, the onload attribute has been neutered properly by replacing it with *onfiltered*, which is an invalid tag. The JavaScript is no longer loaded and executed after the page load.

~~DIS~~INFECTING WITH AJAX

Once the user reads the email, the JavaScript code of the worm begins executing via the unfiltered onload handler. The worm utilizes AJAX (Asynchronous JavaScript and XML), which is another first. Now running under the context of yahoo.com and the currently logged-on user session, the worm has the ability to parse the web page and make the same HTTP queries as if the user had clicked on items in the webmail interface.

Smartly, the worm uses AJAX for the HTTP queries. If the worm had not used AJAX, any HTTP queries would have resulted in another page loading, which would be more likely to be noticeable to the user, as well as putting the calling JavaScript code out of scope.

By using AJAX, the worm can issue multiple HTTP queries in order to find email addresses and send itself all under the covers without changing the page.

JavaScript is global in scope across an HTML page, so script in one block has full access to variables and functions in other blocks or included JavaScript files. JS.Yamanner@m takes advantage of this in order to perform some of its actions.

For example, the first thing JS.Yamanner@m does is to determine which server is being utilized. *Yahoo!* serves the web application from many different servers. The URL is stored in a variable called url0, and from this variable JS.Yamanner@m can parse out the domain.

To collect email addresses, JS.Yamanner@m sends an HTTP query as if someone had selected the QuickBuilder functionality in *Yahoo! Mail* using AJAX so the page does not refresh. QuickBuilder is a *Yahoo! Mail* feature that searches all your mail in selected folders for any email addresses that are not already part of your address book. The purpose of this feature is to allow you to build your address book quickly.

JS.Yamanner@m takes advantage of this feature to get *Yahoo!* to find viable email addresses. JS.Yamanner@m requests the first 100 (alphabetically by folders) email addresses in all folders and parses these for any that match @yahoo.com or @yahoogroups.com. In addition, the worm attempts to filter out the user and sender's email addresses, thus preventing the threat from resending itself back to the user. The worm obtains the user's and sender's email addresses from form fields already populated by the *Yahoo! Mail* application.

With a list of viable email addresses, JS.Yamanner@m sends an AJAX HTTP request to compose a new message, but doesn't actually use this request to send the message. Instead, this request is used merely to generate a new 'crumb', which is similar to a session-tracking cookie, but a form value within the page.

JS.Yamanner@m then sends a second AJAX HTTP request with a variety of POST variables set in order to forward the open message (which contains itself) to the list of email addresses discovered via QuickBuilder. One email address is set as the 'To:' email address and the entire list is set as the BCC: field. Since this is a forwarded message, the From: address will be set to whoever opened the infected message. The subject line is set to 'New Graphic Site' and the message body is set to 'Note: forwarded message attached'.

JS.Yamanner@m also needs to set a variety of administrative values, including a tracking number for the message being forwarded by parsing the HTML page, the crumb value, and it needs to set a parameter so the message isn't saved in the Sent folder.

When the message is received, at the bottom of the chain of forwarded messages, the infected HTML attachment will automatically be rendered and contain the text 'Please wait while loading the site'.

REAPING THE HARVEST

Few users will actually see the content of the message since as soon as the page is loaded, the worm executes and after the worm forwards itself to further targets, JS.Yamanner@m then calls *window.navigate* to redirect the page to another website (www.av3.net), along with a variety of GET parameters.

The purpose of the redirection wasn't completely clear at first. The redirection included GET parameters which appear to have been used for debugging purposes to show the Sent folder view in *Yahoo! Mail* after the worm finished sending itself, but before being released, the yahoo.com domain was replaced with www.av3.net. Also, appended to the parameters was the list of email addresses, which would not be needed when displaying the Sent folder view.

While many assumed that the purpose of this last step was to harvest email addresses and send them to av3.net, av3.net has existed for a long time and is referenced and utilized on a variety of websites. Av3.net hosts multimedia content commonly used on *MySpace*.

In addition, the form of the GET parameters with unnecessary *Yahoo! Mail* parameters didn't quite add up. Furthermore, the author could easily have used AJAX instead of redirection. Perhaps the website was simply a red herring or an attempt at implicating someone else.

After some investigation, we were able to determine that av3.net was owned by the author of the worm and thus, email addresses were harvested. The unnecessary GET parameters were likely added so that the harvested email addresses weren't visible in the URL box. The actual harvesting of the email addresses does not appear to be done by any back-end scripting on the site, but perhaps just parsing of the standard web access logs. Another reason AJAX or a dummy site wasn't used is perhaps because the author desired page views for generating income via hosted advertisements.

RISE AND DOWNFALL

Because the worm redirected automatically to another site that hosted a hit counter, we were able to track infection rates and also get a glimpse of who was infected.

JS.Yamanner@m infected close to 200,000 users between 11–12 June 2006, before dying out due to the vulnerability being fixed. The number of hits is likely to be higher than the number of infected users since a common reaction of a user, when they saw their browser redirected to the av3.net website, would be to hit the back button. But doing so would just cause the infection to execute again and then redisplay the av3.net site. According to the web stats, infected users included a variety of governments and large corporations including financial institutions.

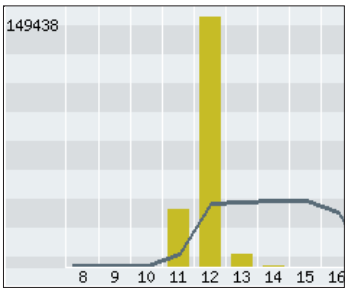

Figure 1: Number of infections per day.

While the worm spread quickly due to the use of a vulnerability in *Yahoo! Mail*, the worm was eradicated immediately as soon as *Yahoo!* patched the vulnerability. At that moment the worm could no longer spread and the hits on av3.net stopped (see Figure 1).

Web applications such as *Yahoo! Mail* have a distinct advantage over client applications since as soon as a vulnerability is patched, all users are protected immediately. In contrast, email applications such as *Outlook* suffer from the fact that not all users upgrade or patch their installations straight away and therefore remain vulnerable to similar attacks.

The country most affected by JS.Yamanner@m was the United States, which is no surprise considering the popularity of *Yahoo! Mail* within the US market. The second most affected country was Iran. The reason for this became clear when we tracked the infection back to 'ground zero' and managed to obtain some information about the author himself.

GROUND ZERO

The worm simply forwards an existing message. However, forwarding the message means that the original headers can be found on previous messages.

After obtaining a few samples, we were able not only to determine the original infection, but also to create interesting relationship trees demonstrating how the worm spread from one user to the next and how users were interconnected.

The diagram shown in Figure 2 is a partial branch. The blue ellipses represent *Yahoo! Mail* groups and the squares represent individual *Yahoo!* email addresses. The red circle on the right is the first infection.

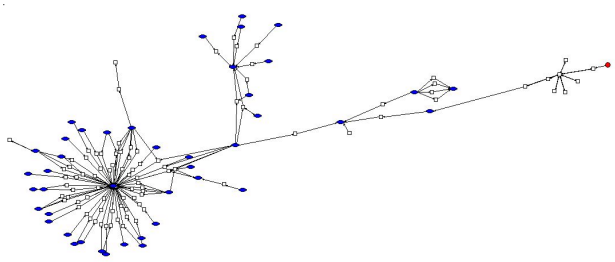

Figure 2: A partial branch of how infected users are connected.

After some investigation of the originating addresses, it seems that the author may have been Iranian, or at least shared interests with and had connections with Persian speakers. Thus, it was no surprise to see that the second most affected country was Iran as the first person to be infected likely had contacts with people living in Iran.

Further confirmation of the worm's country of origin appeared when the author sent an email to anti-virus companies in search of employment.

```
"… Finally I should mention that I don't like to
disturb no one. Since I live in iran and taking a Job
in good computer companies is very hard (becaue
getting Visa is very hard from US) I just want to
prove that I have some abilities in web programming.
And I like to work with professional team like you if
there is any way to do that…"
```

The final piece of data came when attempting to get the site shut down and at the same time determine if it was an innocent party. At first, no contact information was available on the site, so as part of normal procedure we sent a message to the upstream provider.

Later, however, we found that the page had been modified and that it included contact information. A message sent to the address given on the web page was met with a reply that confirmed that the author of the worm owned the site and it even came with a full name. Interestingly, the data indicates that the author has spent time living in Canada. Of course, this could just be someone attempting to implicate someone else. We won't know unless the authorities actually arrest the author.

CONCLUSION

JS.Yamanner@m was not only the first webmail worm, but it demonstrated how web applications are equally susceptible to vulnerabilities. More so, JS.Yamanner@m demonstrates the impact of AJAX and how AJAX can be harnessed to develop more efficient web application worms.

While similar worms will also generally require a vulnerability in the web application, web application vulnerabilities are not rare. Fortunately, as soon as the vulnerability is patched, the worm stops spreading.

VIRUS ANALYSIS 2

STAR WHAT?

Dr Vesselin Bontchev
FRISK Software International, Iceland

On 30 May 2006, a short message appeared on the *Kaspersky Lab* web log [1], announcing the appearance of a new macro virus for *StarOffice*. The log read:

> 'I came across something interesting today, a macro virus which we've named Virus.StarOffice.Stardust.a ... But if you look more closely at the name, you can see why I'm interested: Stardust is a macro virus written for StarOffice, the first one I've seen. Macro viruses usually infect MS Office applications.'

Now, we all know that anti-virus researchers all over the world are overloaded by the continuous glut of new malware and that the constant stress can cause them to make mistakes. And in such a quick-and-dirty publication as a blog, one cannot expect the author to have done extensive research and fact-finding before posting such a short message. Still, how many mistakes can be found in the message above?

To begin with, the name StarOffice is unnecessarily restrictive. *StarOffice* is the name of a commercial product manufactured by *SUN* and based on the code base of *OpenOffice* – an open source product [2]. Any malware supposedly written for *StarOffice* would work under *OpenOffice* too.

Second, OpenOffice is too generic to be used as a proper malware platform name. According to the CARO Malware Naming Scheme [3], one should use either the language in which the malware is written (i.e. StarBasic in this case) or the application for which the virus is specific (e.g. *StarWriter*).

Almost half a decade ago, the author of this article did some research into the 'virusability' of this platform. Back then, *OpenOffice* supported a single macro language, called StarBasic, which was somewhat reminiscent of VBA but the two languages were not compatible. *OpenOffice* was able to open *Microsoft Office* documents – including documents that contained macros – but during the conversion process the macros were simply stripped from the document. In addition, StarBasic did not seem to have the concept of 'self' – it didn't seem possible for a StarBasic macro to say 'copy myself from here to there'.

Of course, that did not mean that viruses for the platform were impossible. The language was powerful enough to implement several kinds of them without the capability of copying macros from one document to another. For instance, it was possible to implement mass-mailers, it was possible to implement overwriters, and it was possible to implement viruses that copied the *data* of the target document to themselves and then overwrote that document – pretty much like the Jinni.A virus does in *MS Excel* [4]. *OpenOffice* also conveniently allowed any macro subroutine to be attached to any event – such as a menu selection or a keypress. Still, it didn't seem likely that such viruses would have a significant chance of spreading and becoming a problem.

Sadly, things have 'improved' a bit since then. Nowadays, *OpenOffice* supports several different languages: StarBasic, BeanShell, Python and JavaScript. They are all *script* languages, however – not *macro* languages – which leads us to another error in the original announcement. Whatever the new thing was, it couldn't have been a macro virus; at best, it was a script virus.

In addition, the StarBasic language has become much more powerful too. We became aware long ago that full-featured viruses for it were possible; see for instance [5]. So, the news that someone had, supposedly, finally written a virus for it was hardly a surprise. For more information, though, we needed to analyse a sample.

NOT ANOTHER FIRST

The sample arrived in an archive containing three files. One of them was in a subdirectory named 'Dropper', so it was assumed that this contained the first generation of the virus. The other two were, supposedly, infected documents.

The files that contained the samples all had the SXW extension – in other words, they were *StarWriter* documents. Initially, we did not enjoy the prospect of having to reverse-engineer yet another undocumented file format (we have had more than enough undocumented file formats used by the various *Microsoft* products), but then we remembered that *OpenOffice* uses a different concept. The format *really* is open – the 'document' is essentially a ZIP archive, containing various XML files in various subdirectories. Indeed the idea is so good that *Microsoft*, allegedly, intends to steal/borrow/innovate it in the next version of *Office*.

So, it really is easy to take apart the contents of an *OpenOffice* document. It took us just a few seconds to find the XML file that contained the 'macro virus' code.

Now, the author of this article does not claim to be a StarBasic expert, but things started looking fishy to him immediately. The code contained three subroutines: AutoInstall, mygame and InstallGlobalModule. The first of these just called the second. This already looked strange, because 'AutoInstall' is not a special subroutine name in StarBasic (e.g. as 'AutoOpen' is in WordBasic or 'Auto_Open' is *Excel*'s version of VBA). But let's assume

that the subroutine does manage to get invoked somehow – perhaps by having been attached to a frequently-called event or something.

The next problem was in the subroutine mygame. It began with the comments:

```
&apos;******************************
&apos;******* OOo.Stardust  *********
&apos;*** (c)by Necronomikon[DCA] ***
&apos;******************************
```

and ended with what appears to be a call to InstallGlobalModule:

```
&apos; InstallGlobalModule()
```

One does not have to be a StarBasic expert to realize that there is no way the above could work.

As is obvious from the first few lines, the sequence ''' is clearly a comment – a notion supported by the fact that it resolves to the ' (apostrophe) character in XML and in most Basic dialects this character is used to indicate that what follows (until the end of the line) is comment. So the call to the InstallGlobalModule subroutine (which seemed to contain the bulk of replication code) is commented out – i.e. the replication is never invoked.

A discussion with some other CARO members (e.g. [6]) confirmed that the thing was unable to replicate – i.e. it was not a virus, but an intended. Here we have another error in the original announcement, which claims that it is a virus. Admittedly the announcement also said 'Stardust is the first virus ... which is *theoretically* capable of infecting StarOffice…', so it could be argued that it was talking about a potential virus, not a real virus. However, the rest of the announcement quite unambiguously calls the thing a virus and it never states explicitly that the malicious code simply doesn't work.

As we shall see in the next section though, *Kaspersky Lab* was far from alone in making this mistake.

AMAZING INCOMPETENCE

The alleged author of this thing, a malware writer who uses the handle 'Necronomikon', is well known to us from the macro malware world. Well known for his incompetence and ineptitude, that is. He is the author of several macro intendeds, like Delay.A, Gamor.A, Hilite.A, Hilite.B, Hilite.C and Hilite.G.

For those who came late to the party, an intended is a program written with the obvious intent to make a virus, but which is too buggy to replicate. It's a double joke on its author – not only was he too incompetent to create a working virus (a rather trivial task, especially in the macro and script worlds), but was also stupid enough to release it without trying to run it (otherwise he would have noticed that it doesn't work).

But the commented-out call to the replication routine is not the only problem for this piece of malicious code. Even if the comment is removed, it cannot be made to work. The observant reader would notice that the subroutine is then invoked without any parameters. But the declaration of this subroutine, just a few lines below, clearly indicates that it is supposed to take one mandatory and two optional arguments:

```
Sub InstallGlobalModule( ByVal cGlobalLibName As String,_
   Optional cDocumentLibName,_
   Optional stardust )
```

The code after that *relies* on the presence of the first argument and tries to take some reasonable action if the two optional ones are missing:

```
If IsMissing( cDocumentLibName ) Then
   cDocumentLibName = cGlobalLibName
EndIf
If IsMissing( stardust ) Then
   InstallGlobalModule2( cGlobalLibName, cDocumentLibName,
BASIC_MODULE )
   InstallGlobalModule2( cGlobalLibName, cDocumentLibName,
DIALOG_MODULE )
Else
```

Apparently, the subroutine is supposed to perform some kind of generic module-copying function and to copy a given module to different places, depending on how the subroutine is invoked. But since it is never invoked correctly, we can't know that for sure. It looks as if the malware author has taken this subroutine from somewhere without really understanding what exactly it is supposed to do and how it is supposed to be called.

But it gets even worse. Apparently, the procedure InstallGlobalModule2 is not part of the standard *OpenOffice* installation [7] – and it is not present anywhere in the code of the 'virus', either. So, even if the replication function had been invoked correctly, it would fail to work because it refers to a non-existent subroutine.

The code immediately after the one quoted above also calls a function named 'DoesModuleExist' – which is not part of the standard *OpenOffice* installation either, and the implementation of which cannot be found anywhere in the code of the sample we received.

In other words, there was absolutely no way the code we were looking at could be a virus. It didn't just have some trivial bug that could be fixed, either – it looked more like a random collection of code, whose author didn't really understand what that code was supposed to do.

Then we turned our attention to the so-called 'dropper'. Maybe, we thought, the dropper has the capability to infect

documents once, but the code is incapable of propagating for more than one generation – e.g. because some important part of it isn't copied around after the first time. However, even a cursory inspection of the code of the 'dropper' proved that this couldn't be the case. Furthermore, there was absolutely no way the 'dropper' could have produced the code in the other two samples.

When comparing the code of the 'dropper' with that of the other two samples, we discovered only two differences. One of them was that they began with two different sets of declarations. The 'dropper' begins with:

```
Dim lAutoInstall as Boolean
Dim Url As String
 dim myFileProp as Object
```

while the code in the 'samples' begins with:

```
Const GLOBAL_LIBRARY = True
Const DOCUMENT_LIBRARY = False
Const DIALOG_MODULE = True
Const BASIC_MODULE = False
Dim lAutoInstall as Boolean
```

Now, there is no obvious reason why the code should change like this. There are no operators that attempt to modify these lines at the beginning. However, in VBA, the source of the module often begins with a series of attribute statements. These don't generate any p-code and can change when a module copies itself from a document to the global template (as well as under several other circumstances), so we decided to give this difference the benefit of the doubt – maybe it was produced automatically somehow by *OpenOffice*.

The second difference, however, was clearly put there by the malware author and there was absolutely no reasonable explanation for its absence in the supposed replicants. The difference is that just before the commented-out call to InstallGlobalModule, the 'dropper' contains the following lines:

```
otext=oDocument.text
ocursor=otext.createtextcursor()
otext.insertString(ocursor, &amp;quot;***Stardust***(c)by
Necronomikon[DCA]&amp;quot;,false)
url=converttourl(&amp;quot;http://stardustvx.tripod.com/
SilviaSaint.JPG&amp;quot;) &amp;apos;nice idea from
Slagehammer... ;)
oDocument = StarDesktop.loadComponentFromURL(url,
&amp;quot;_blank&amp;quot;, 0, myFileProp() )
```

These lines attempt to download a file and to display it on the desktop. Since there was no obvious reason for their absence in the replicants, the only reasonable conclusion was that, in reality, we weren't looking at a dropper and replicants of the virus it drops – we were looking at two different variants of non-viral malware.

Now, one incompetent malware author is hardly a surprise – among them, ineptitude is rather the rule than the exception. What *is* surprising, however, is how many supposedly competent anti-virus companies believed his claims and described these two different things as a single virus for *OpenOffice*.

'The Stardust virus doesn't appear to work very well', says *Sophos*'s Graham Cluley [8]. 'Not very well', huh? How about not at all?

'We have a sample of a proof-of-concept macro-virus for OpenOffice.org', writes *F-Secure*'s Sean in the company's blog [9]. OpenOffice.org is a website and this thing is neither macro, nor a virus. It's an intended script.

'Type: virus' has been written by a researcher from *CA* in their description of this thing [10]. Nope, it ain't.

'SB.Stardust.A!int is a proof-of-concept macro virus for Sun StarOffice documents', according to *Symantec*'s description [11]. It's not a virus, folks! Well, at least *Symantec* gets points for appending 'int' at the end of the name, suggesting that it's an intended. But why not say so clearly in the description? (*McAfee* uses a similar approach – '.intd' is used in the name, but the author of the description is shy of stating clearly that the thing doesn't really work at all.)

'It is the first time that the experts detected a macro virus called "Stardust" in Internet, which takes advantage of the Office-Suite OpenOffice for its attack. The moment the user opens the document template, the script that was written in StarBasic will infect all the following documents', claim the researchers from *Avira* [12]. Guys, your so-called 'experts' aren't.

SoftWin's scanner calls the thing 'Worm.StarOffice.Stardust.A'. A worm? Now, I know that the experts can never agree on the exact definition of a worm, but it certainly isn't a horribly buggy piece of non-working code that doesn't even run, or replicate, let alone replicate over networks.

Researchers at *Trend Micro*, seemingly, couldn't make up their minds about what the thing really does [13]. On the one hand, their description says: 'Once an infected document is opened, it downloads and opens an image of an actress from a certain website. It then proceeds to infect other StarOffice/OpenOffice Suites document files', which is clearly false. But then it adds: 'However, due to some errors in its code, it cannot perform its infection routine'. Good job! But why, then, does the description state: 'In the wild: Yes'? In the wild? No way! In the wild imagination of the journalists, maybe.

Doesn't anyone analyse the virus samples they receive these days?

And that's just the anti-virus companies – the ones that are supposed to know better. What can we say about the popular computer press that has never been distinguished with competence in this (or, for that matter, any other) area? As anticipated, *PCWorld*, *ZDNet*, *News.com*, *InfoWorld*, and all the rest have jumped on the 'Stardust virus lands on *OpenOffice*' bandwagon.

In fact, even the official response of OpenOffice.org to this 'threat' [14] refers to it as a virus. Although the statement reads: 'technically, it is not even a virus, as it is not self-replicating', it also goes on to say 'with OpenOffice.org's default settings, it cannot spread without user intervention'.

Folks, the only kind of user intervention that would make this thing spread would be taking the module it resides in and copying it manually elsewhere. But, with that kind of user intervention, even the 'Hello world' macro is a virus. One would have hoped that at least the guys at OpenOffice.org are capable of reading and understanding StarBasic – but apparently not.

STAGE THREE

At this stage, a private communication [15] from Gabor Szappanos turned my attention to the fact that the author of this malware seems to have borrowed the InstallGlobalModule subroutine from a publicly available package of macros [16].

Indeed, if you look inside this package, you will find not only the InstallGlobalModule subroutine but also implementations of the missing InstallGlobalModule2 and DoesModuleExist. It became clear that the author of the thing had found this library, and had worked out that it seems to have the ability to copy modules from place to place, but that he had been unable to understand how it works exactly, and why.

The malware author has tried to copy the relevant parts from the library, but since he was unable to determine which parts are relevant and how they are supposed to be used, he has messed up. Badly. Being an impatient kid, however (aren't they all?), he has rushed to send his creation to all the anti-virus companies in order to get his 15 minutes of fame. And he has received more fame than he deserved, mostly due to the incompetence of the popular media and of said anti-virus companies.

But, apparently, the author of this malware had a nagging feeling that his creation wasn't quite perfect (to say the least). So, he continued to 'improve' it and has released yet another (third) variant. Unfortunately, he forgot, once again, to test whether the thing actually works – or at least to read the documentation of StarBasic – because the third variant is again an intended.

The main difference this time is that the author has also lifted the function DoesModuleExist from the public macro library mentioned above. He has also changed all calls to InstallGlobalModule2 to just InstallGlobalModule – because he still hasn't taken the former subroutine from the macro library. Unfortunately, these calls are all inside a subroutine named InstallGlobalModule too – which would normally lead to infinite recursion and a crash. If that subroutine was called at all, that is. Because it isn't. There isn't even a commented-out call – nothing at all.

THE NAME OF THE WEED

So we still don't have any viruses for *OpenOffice*, no matter what the hysterical media and the incompetent anti-virus companies are claiming. Still, we do have some malware for it (three intended variants), so we needed a name for the platform in the CARO Malware Naming Scheme, as well as a family name for the trinity of variants.

Strictly speaking, StarBasic is a script – not a macro language, since it is just ASCII text and is not tokenized or compiled in any way. So, we felt that StarBasicScript (or SBS for short) would be the proper platform name. However, in the name of brevity, we decided to use just StarBasic (or SB for short) instead.

As for the family name, there were proposals for 'Bulldust' or 'Dustar', but the name 'Stardust' – picked by the malware author – was so widely hyped by the media, that we decided to go with it, in order to avoid additional naming confusion.

So, the full names of the three variants are:

intended://StarBasic/Stardust.A

intended://StarBasic/Stardust.B

intended://StarBasic/Stardust.C

CONCLUSION

OpenOffice is a virusable platform – it is perfectly possible to write a virus for it. At the time of writing this article (June 2006), there were only three non-working attempts at a virus written by somebody who obviously has more time than brains.[1]

Considering this, we would suggest that 'Necronomikon' changes his handle to 'Necromoron' – from the Greek necros ('dead' – as in 'brain-dead') and the English moron ('dolt'). We feel that the latter handle reflects his mental abilities more aptly.

REFERENCES

[1] Stardust – a macro curiosity. http://www.viruslist.com/en/weblog?weblogid=187738337.

[2] StarOffice FAQ, Question #13, available from http://www.sun.com/software/star/staroffice/faqs/technical.jsp#q_13.

[3] Current Status of the CARO Malware Naming Scheme, available from http://www.people.frisk-software.com/~bontchev/papers/naming.html.

[4] Bontchev V. The Three Faces of VBA – Part 2. Virus Bulletin, February 2005, pp. 4–6.

[5] Rautiainen S. OpenOffice Security. Proceedings of the 13th Virus Bulletin Conference, 2003, pp. 51–57.

[6] Raiu C. Kasperksy Lab. Personal communication.

[7] FitzGerald N. Personal communication.

[8] First virus for StarOffice poses no serious threat. http://www.sophos.com/pressoffice/news/articles/2006/05/stardust.html.

[9] OpenOffice and Ziggy Stardust. http://www.f-secure.com/weblog/#00000893.

[10] http://www3.ca.com/securityadvisor/virusinfo/virus.aspx?id=56047.

[11] http://www.symantec.com/avcenter/venc/data/sb.stardust.a!int.html.

[12] OpenOffice is vulnerable – Avira warns against the first macro virus. http://www.avira.com/en/security_news/openoffice_is_vulnerable_-_avira_warns_against_the_first_macro_virus.html.

[13] XML_DUSTAR.A. http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=XML_DUSTAR.A.

[14] Statement on the Proof of Concept Macro Virus. http://www.openoffice.org/press/statement-proof-of-concept-virus.html.

[15] Szappanos G. VirusBuster. Personal communication.

[16] Library Installer. http://kosh.datateamsys.com/~danny/OOo/Experiments/obsolete/LibraryInstaller-2004-03-09-01.sxd.

FOOTNOTE

[1] *While this article was being prepared for publication, a real working virus for the OpenOffice platform appeared – SB/Starbugs.A. It is written by a different, less inept virus writer and works under all OpenOffice applications that can run StarBasic.*

FEATURE

DIAL M FOR MALWARE

Tomer Honen and Alexey Lyashko
Aladdin Knowledge Systems, Israel

Not so long ago our parents were telling us not to sit too close to the television set. Today, kids are watching music videos on their cell phones, with their eyes two inches away from the screen. When we were kids, we sent notes to each other in class and risked being thrown out of the lesson if caught. Today, children send each other SMS messages and no one's the wiser.

Many of us are concerned about our health and our privacy, yet we carry around devices that expose us to radiation (some say harmful, some say mundane) and which allow third parties to triangulate our position in the world and listen remotely to everything we say – yes, even if the phone is turned off.

Only a few years ago, mobile phones were just what their name implied – phones that could be carried around and which could receive calls anywhere, even while the user was away from home (they also looked like and weighed as much as miniature freezers, but that's beside the point).

Today, many of these devices incorporate a stills camera, a camcorder and a satellite navigation system, and can run games and other utilities – one application can even handle phone calls [1]! Many of today's mobile phones use a complex operating system capable of accomplishing various tasks; in fact, some mobile devices are primarily PDAs (Personal Digital Assistants). The ability to handle phone calls is merely a secondary feature.

THE PRICE OF POPULARITY

The *Symbian* operating system is to mobile phones what *Windows* is to PCs. It is one of the most common operating systems for these platforms and as such it enjoys a wide variety of commercial and free open-source applications that are developed daily. This popularity, however, does not come without its share of problems, namely viruses. Since a virus writer's goal is to infect as many targets as possible, the *Symbian* OS is the most obvious mobile platform for virus development.

While the current number of viruses developed for this platform is far from staggering – a little over 220 in all (compared to tens of thousands of viruses targeting PCs), one must bear in mind that the technology is still young and the more advanced mobile phones are still quite expensive. However, experts predict that the coming years will see a substantial increase in mobile phone sales. According to *Gartner*, during the first quarter of 2006 an astounding total of 224 million units were sold around the world, an increase of 23.8% from the same period last year. Based on their predictions, close to a billion units will be sold by the end of this year!

PROPAGATION

At the present time there is little innovation among the threats targeting mobile phones. Of the 220-odd viruses out there, only a few are completely original. The rest simply keep reusing and recompiling the same code over and over again.

The first viruses of this kind used Bluetooth as their main method of propagation. While the technology offered a quick and a relatively anonymous way of transmitting viruses to others, it relied on these users being moderately close to one another – usually up to 10 metres (providing there were no obstacles along the way).

While Bluetooth is still used sporadically by malware, today most threats are downloaded directly from the Internet or sent manually by malicious users. A few threats use MMS (Multimedia Messaging Service), which is similar to email in that it allows users to send out all types of files, not just plain text. In fact, viruses that are capable of sending themselves via MMS enjoy the same advantages as those that spread by email, which means they have the ability to send many copies of themselves to other users, thus propagating constantly. Add to this the fact that most people using MMS-capable devices do not have any anti-virus protection and you have a potential epidemic.

Mobile phones, however, are not completely exposed as they are devices that use one of a handful of available solutions. These are similar to desktop-based anti-virus programs. Provided that users keep updating the software's database with the latest mobile phone virus signatures, they will be safe. However, this solution could be problematic for some users.

For one, such applications take up valuable memory. Even PC-based anti-virus solutions can prove cumbersome for some desktop computers. On a mobile phone, where memory is quite limited to begin with, this issue is more obvious. In addition, while many PCs can stay online virtually indefinitely and receive all the updates they require as soon as they are available, mobile devices cannot; maintaining an Internet connection can be expensive. Even if that is not a problem, the level of radiation generated by these devices over long periods of operation may be troubling to some. This makes updating the installed anti-virus solution regularly a chore – and a costly one at that. Users are likely to remain unprotected from new threats for quite a while before a solution is applied.

When dealing with viruses one has to be protected around the clock; in many cases an MMS virus is likely to reach users faster than its remedy simply because it is independent of any user interaction. But why stop there? Any mobile phone capable of connecting to the Internet is exposed to numerous risks other than viruses – such as phishing attacks, spam and even spyware [2].

Will the current generation of mobile phone anti-virus solutions be able to protect users from all of these threats? Highly unlikely.

THE THREAT

As mentioned above, despite the fact that there are quite a few types of *Symbian* malware out there, they can be separated roughly into around five or six families, each using very similar source code (in terms of structure and functions). When a current generation of mobile malware is installed on a victim's mobile device, it starts sending copies of itself to all the contacts it can find. It may also send private information found on the system. Of course, another unwanted effect of the virus is that the user's monthly bill from the cellular service provider may be quite substantial as well.

A good example of more generic, but potentially damaging mobile malware is Comwarrior. This virus targets *Symbian* OS-based mobile devices and demonstrates all of the above behaviours. It also distributes itself via MMS. In addition, it has Bluetooth spreading capabilities which it uses to infect devices located nearby. It is usually quite a common practice to include two or more types of virus in the same SIS package [3]. Upon execution, one of the dropped viruses will be responsible for distribution via Bluetooth, another via MMS, while the third executes a damaging payload, etc.

There are several proofs of concept that are able to distribute themselves across different platforms. The well-known Crossover virus is able to replicate itself between the *Pocket PC* and the *Windows* operating system, for example. Although malware like this has not been met in the wild yet, the door has been opened and it can only be a matter of time before real malware of this kind, not just a 'lab-virus', is released to the world. While the previous example may not specifically affect mobile phones using the *Symbian* OS, a cross-platform virus is feasible for these devices as well.

It all comes down to a popularity contest of sorts. As soon as mobile phones become more common (one billion units a year sounds about right) they will draw the attention of more and more malicious code writers looking for a challenge – or worse, profit.

The full potential of malware targeting mobile devices has not yet been realized – we probably have not even seen the tip of the iceberg. The next threat could create the following scenario: Ed, an employee at a high-tech company receives an MMS with an attached SIS package while on the way home from work. The text message claims that the file is a critical system update, a freeware game, or anything else that could coax a user to run the application. He can't reject the opportunity to install some free software or a critical update on his system – especially in an age where many users are not aware of such threats (that receive nearly no media attention at all).

Once the program has been installed, Ed sees no difference in the device's behaviour. Meanwhile, however, personal data such as his contact list, organizer records etc. is being collected. This could also include photos taken with the device's camera when Ed, his wife and their kids were on vacation, or work-related documents and SMS messages.

Current generation *Symbian* threats can already perform some of these actions, so let's take it a step further: when Ed finally gets home, he says 'hi' to the family and then connects to his office PC, since he forgot to answer a few emails. He places his mobile phone on its cradle to synchronize messages with his PC's email applications. This is where things get interesting; the virus detects the connection to Ed's PC and carries out the rest of its payload. It drops several files onto the PC without Ed's knowledge and executes them in the background. Ed's computer can now be infected by spyware, a backdoor trojan or some other malicious program that may eventually find its way to his PC at work.

Although this is a fictional scenario it is not far-fetched and could actually happen, at least theoretically. Only time will tell. Right now mobile phones are becoming more and more advanced. We are not too far from the day where mobile threats will be as sophisticated as their PC counterparts.

Surprisingly enough, one does not have to look far to find a solution that would protect users against this kind of threat: a suitable solution is already used by ISPs to protect PC users.

THE BEST OF BOTH WORLDS

Since desktop anti-virus solutions do not provide complete protection against online threats, many corporate networks employ a firewall to block illegal intrusion attempts. Many also install gateway content security solutions that are capable of scanning traffic as it is downloaded, thus complementing both the firewall and the desktop anti-virus and providing a much better chance of avoiding malware altogether.

The first two solutions can usually be installed by experienced users or technicians and both can easily be downloaded from the Internet, sometimes free of charge (albeit with reduced functionality – which should still be enough for many users). However, gateway content security requires a lot of resources. It requires certain specialised equipment, an expensive application and – most importantly – constant supervision by an experienced system administrator. For the average user this is not a reasonable solution.

A desktop-based anti-virus solution is usually the most common, affordable solution. However, the human is the weak link in the chain here as few users actually bother to update their software regularly. Many users would like to know that their systems are protected without the hassle involved with micro-managing the program.

A growing trend among Internet Service Providers (ISPs) helps such users protect themselves better by eliminating the need for constant human interaction. These ISPs provide users with their own gateway-like filtering system that requires no maintenance on the user's part.

Simply put, the system scans content as it is downloaded by the user. Malicious content is blocked before it can cause any harm and the user is informed about the situation by a message displayed in the Internet browser's window. For a small monthly fee users can be certain that they are protected against all Internet-borne threats without being bothered by daily updates, obscure threat alerts and various software issues. Desktop anti-virus solutions can then be used solely for the purpose of scanning CDs, flash drives and other portable media which cannot be scanned by the ISP's gateway filtering. From the user's point of view, this is a simple, yet highly effective solution.

Why not do the same for mobile phones then?

This realization has spurred a new trend among mobile phone service providers – gateway content security for their customers. In a similar manner to the solution described above, the gateway's content security takes place between the Internet and the service provider's network.

While this system complements the device-based solution, the provider's solution offers much more than simply blocking viruses. In fact, why not block phishing, spam, PC malware and spyware altogether? While the latter two threats do not (yet) pose a direct threat to the mobile device itself, they may be transferred to a PC at a later stage and cause much havoc.

THEN AND NOW

Computer history is filled with naysayers, be they those who say that 'there is a world-market for maybe five computers' (Thomas Watson, Chairman of IBM, 1943) or the few individuals who proclaimed there was no way viruses could propagate by email (usually computer virus experts responding to users' fears over the Good Times hoax [4] around 1994).

It is easy to dismiss mobile viruses for so many reasons; the relatively low propagation of the threats and their simplicity from a technical standpoint, the low availability of high-end devices and the seemingly minimal damage current-generation mobile viruses can inflict upon unprotected users. The truth is that similar things were said about computers and computer viruses. There is no such thing as overkill when dealing with malicious content and the old cliché of 'better to be safe than sorry' is always applicable in this case.

When updated regularly, device-based anti-virus solutions provide excellent protection against the few known threats that are currently in the wild (in active propagation). But for all other threats, from those that started circulating before you had a chance to get that latest update to those threats that target your PC, a gateway solution at the service provider's end is, in many cases, as essential as the service itself.

END NOTES

[1] This article was written by two individuals who own (or are owned by) such infernal devices.

[2] Spam and spyware are already a growing problem affecting mobile phones users.

[3] SIS packages are files similar to executable installers on the PC. The have a certain list that instructs them where to extract each and every file located in the package.

[4] Many consider the Good Times (or Goodtimes) virus-warning hoax as a precursor to self-replicating worms that started propagating only a year later. The description of this 'virus' is strangely similar to the Melissa worm and similar threats. More information is available at http://en.wikipedia.org/wiki/Goodtimes_virus.

[*Symbian threats will be discussed in detail at this year's Virus Bulletin conference (VB2006): Dr Vesselin Bontchev will look at the problems associated with Symbian malware classification, and Robert X Wang will take 'a deep look into Symbian threats'. VB2006 takes place 11–13 October 2006 in Montréal, Canada. The full conference programme, including abstracts for all papers, and online registration can be found at http://www.virusbtn.com/conference/.*]

COMPARATIVE REVIEW

NOVELL NETWARE 6.5

John Hawes

The previous incumbent in this post, Matt Ham, made no secret of his opinion of the *NetWare* operating system and the anti-virus products available for it. Though he left the job exactly one month before the review schedule came back round to *Novell*'s network operating system, this may be mere coincidence. Faithfully following the test timetable laid down before my arrival, I resolved to ignore Matt's cynicism and approach the task with an open mind. Wide-eyed and full of wonder, with the prospect of making friends with the gaggle of strange new AV products before me, I headed into the lab.

PRODUCTS, TEST SETS AND PLATFORMS

One of my first tasks for *VB* was to issue a call for products and to announce deadlines for this test. I chose the date of my first day in the job, 3 July 2006, as the vendors' final chance to submit products and virus data updates, with the WildList deadline a few days earlier; as a result, the In the Wild (ItW) test set was compiled using the April 2006 WildList.

Fortunately for me as much as for the submitted products, there were comparatively few new viruses to add to the test set; while quite a few fell from the list, only around 30 had been added since the *VB* collection was last updated. Along with the handfuls of W32/Mytob and W32/Bagle variants, there were a few variations of W32/Feebs and W32/Lovgate, as well as some names that were new both to me and the list – W32/Nugache, W32/Gurong and W32/Rontokbro are yet more mass-mailing worms with some file-sharing exploitation and backdoor functionality thrown in.

I was also thankful that, for this educational first stab at running *VB*'s comparative testing, a fairly limited selection of products was submitted. I knew practically nothing about most of these products – most of their names and reputations were familiar only from previous reviews in this very publication. As the products arrived, in the form of zipped email attachments, links to FTP sites or descriptions of CDs stashed somewhere deep in the *VB* test lab, I could only wonder what delights and horrors lay ahead of me.

The test machine setup gave me my first real challenge – one in which I quickly conceded defeat. The current version of *NetWare*, 6.5, with the latest Consolidated Support Pack, number 5, is also known as *Novell Open Enterprise Server* (with the support pack renumbered 2). My hopes that the installation CD with the support pack pre-applied would install happily on the shiny new hardware in the test lab evaporated quickly, when it decided it could not begin to cope with the hardware configuration or components. With time pressing, I decided to avoid fiddling about with drivers and such, and installed instead on older, more standard machines, using more powerful hardware for clients. These ran *Microsoft Windows XP Professional SP2*, with *Novell*'s *Client 4.91 SP2* installed. This compromise meant that the *NetWare* servers were running rather close to the minimum permitted RAM, but they seemed to handle it without complaint.

With products gathered, test collections in place and all the machines happily networking and reimaging, I was ready to commence testing.

CA eTrust v7.1 for NetWare (InoculateIT engine 23.72.00, 23.72.57)

ItW Overall	100.00%	**Macro**	99.72%
ItW Overall (o/a)	100.00%	**Macro (o/a)**	99.72%
Standard	99.82%	**Polymorphic**	99.89%

Opting to run through the products alphabetically, I started with *CA*'s offering – perhaps an unfortunate decision as it proved the most time-consuming product to test. Installation of the *NetWare* product took the form of a *Windows* installer, with a simple and fairly helpful GUI taking me through the steps of selecting the target machine and the components to install. Updating was a little more old-school, with a selection of virus data and engine updates copied onto the server manually, overwriting the existing files and requiring a simple unload and reload of the software to be picked up (I later discovered a more sophisticated approach was also available).

Once up and running, I found the interface on the *NetWare* console fairly intuitive, with the top half of the screen displaying status and statistical information, and a menu of options below. A scan of the test set was easily set up and initiated, although there was no option to browse files or save paths. The scan presented me with a screen showing nothing but the path being scanned and the number of files processed, incremented in hundreds. Results finally appeared at the end of the scan, and were written to a log with much of the information about the scan crammed into the lengthy filename.

As I came to the on-access test I ran into trouble. While the console interface allowed me to stop and start real-time scanning, and to examine the status (opening a new screen showing numbers and categories of files scanned and infections found), there seemed to be no way of configuring the scanner's behaviour. The default settings were to 'cure' infected files, with no obvious form of logging. Resorting to

On-access tests	ItW		Macro		Polymorphic		Standard	
	Number missed	%	Number missed	%	Number missed	%	Number missed	%
CA eTrust (InoculateIT)	0	100.00%	16	99.72%	1	99.89%	4	99.51%
CA eTrust (Vet)	0	100.00%	12	99.82%	1	99.95%	3	99.84%
Doctor Web Dr.Web	0	100.00%	0	100.00%	0	100.00%	0	100.00%
ESET NOD32	0	100.00%	0	100.00%	0	100.00%	0	100.00%
Kaspersky Anti-Virus	0	100.00%	0	100.00%	0	100.00%	0	99.85%
McAfee NetShield	0	100.00%	0	100.00%	0	100.00%	0	100.00%
Norman FireBreak	0	100.00%	0	100.00%	180	91.24%	12	99.45%
Sophos Anti-Virus	0	100.00%	8	99.80%	0	100.00%	15	99.30%
VirusBuster VirusBuster	0	100.00%	0	100.00%	124	92.59%	25	99.14%

the manual at this early stage, and browsing through some of *VB*'s previous *NetWare* comparatives, I discovered that configuration could only be effected via an interface on the *Windows* client. This I duly installed, and I found myself faced with a multi-tabbed browser-based 'Threat Management Console' interface. After upping my screen resolution so I could see at least most of the page at once, I navigated my way around some rather baffling pages, and eventually managed to persuade it first to 'discover' and then to control the *NetWare* product. With this hurdle out of the way, I found the interface itself to be fairly easy on the brain.

Testing proceeded without further incident, the product handling the test set quite happily. However, since the *InoculateIT* engine is not the default for the product, it does not qualify for a VB 100% award.

CA eTrust Antivirus v7.1 for NetWare (VET engine 12.06.01 12.06.2285)

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

The default Vet engine, while very slightly slower in the throughput tests than the alternative provided, achieved marginally better results in the zoo virus detection, and did just as well in scanning the ItW and clean test sets, earning *CA* its VB 100% award. Switching between the two engines was a simple manoeuvre, involving selecting the appropriate option from a menu; again, while this could be done from the console interface for on-demand scans, the client-based management GUI was required to adjust the on-access component.

Doctor Web Dr.Web for Novell NetWare v4.33.3(.06190)

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

Dr.Web proved a much simpler piece of software, with a large number of tiny virus data files and an NLM copied onto the server and loaded. The console screen presented was a rather murky dark-green-on-black, with a small menu in one corner and most of the screen given over to contact details for the company. The menu itself was simple and logical, with ample configuration options, even offering to detect any jokes I may have had on my machine. On-demand scans were accompanied by a highly detailed information screen.

The product flew through the WildList viruses without difficulty, and did well in the zoo collection too; unfortunately, it claimed one of the clean files was infected with 'Trojan.classic' – an issue which, according to the developers, was fixed less than 24 hours after the close of entries for the test, but one which was sufficient to deny *Dr.Web* the coveted VB 100% award this time round.

ESET NOD32 version 1.1640

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

The simplest of the products by far, *NOD32* provided me with only six files, two of which were basic user guides, while a third formed the EULA. The other three files, once copied to the *NetWare* server, provided a command-line scanner, which merrily zipped through the test set, and an on-access monitor, again with all options passed in as command-line qualifiers. Display and logging were simple and effective, although logs were afflicted with the common problem of truncating long filenames, while speed and detection rates were exceptional.

NOD32 takes the VB 100% award easily in its stride; the only other flaw I could find was on the help screen, entitled 'NOD32 Antivirus System for Nowell Netware'.

Kaspersky Anti-Virus for Novell NetWare v5.60.01

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

Kaspersky Anti-Virus is another product installed from the *Windows* client, with standard *Windows* installer dialogues to select server, apply licences etc. Along with the scanner, a ConsoleOne snap-in and web management tool are offered as optional modules; at least one is required as no control at all is possible from the *NetWare* console. A screen is available on the *NetWare* server, with some statistics and status information, but this is purely for display. The option to add a line to the Autoexec.ncf, causing the product to be loaded on restart of the *NetWare* server, is also offered during the install.

I used the ConsoleOne snap-in which, like all ConsoleOne experiences, tended to suffer moments of extreme slow motion. The snap-in provides tree entries for on-demand, on-access and updating jobs, each with a properties page offering copious configuration options. Scans were simple to set up and run, and the interface fairly intuitive and usable.

With almost total success in the virus scans (the only files missed were in archives, not scanned by default on access to save resources), and no false positives, *Kaspersky* wins yet another VB 100% award.

McAfee NetShield for NetWare v4.6.3

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

Once again going for the *Windows* installer approach, *McAfee* has opted also to provide its own client-side interface. The installer slowed things down by demanding the Java Runtime Environment be available before it would consent to continue; with this in place, the software for the *NetWare* server and the *Windows* GUI installed quickly and easily.

A console screen on the *NetWare* server provides information but no control other than totally unloading the scanner. The *Windows* GUI requires a password to access it, which brought testing to a halt once more – I wrongly assumed it wanted the password for the *NetWare* server, when in fact it had its own, presumably as some kind of second-line licensing technique.

Once access was gained, tweaking the settings was straightforward and speedy. Scanning over the test sets proceeded without incident, and the *McAfee* product, while somewhat on the slow side, was admirably thorough, detecting everything that was thrown at it and deserving its VB 100% award.

Norman FireBreak v4.76.2325

ItW Overall	100.00%	**Macro**	100.00%
ItW Overall (o/a)	100.00%	**Macro (o/a)**	100.00%
Standard	99.48%	**Polymorphic**	91.24%

Norman's *FireBreak* also installed from *Windows*, demanding a lengthy licence key before proceeding. It also required the root of the SYS drive of the *NetWare* server to be mapped to a local drive letter on the client. The installation process mentioned a ConsoleOne-based interface, which I was unable to locate on completion; however, it provided a server console interface too.

There were, in fact, two console screens: the first was a monitor packed with information about real-time scanning, while the other was half-empty, with just a small menu in the top left-hand corner. This provided further menus within menus, all arranged in a fairly straightforward and sensible fashion, allowing me to configure the test scans without difficulty.

On-demand tests	ItW		Macro		Polymorphic		Standard	
	Number missed	%	Number missed	%	Number missed	%	Number missed	%
CA eTrust (InoculateIT)	0	100.00%	4	99.72%	1	99.89%	4	99.82%
CA eTrust (Vet)	0	100.00%	12	99.82%	1	99.95%	1	99.96%
Doctor Web Dr.Web	0	100.00%	0	100.00%	0	100.00%	0	100.00%
ESET NOD32	0	100.00%	0	100.00%	0	100.00%	0	100.00%
Kaspersky Anti-Virus	0	100.00%	0	100.00%	0	100.00%	0	100.00%
McAfee NetShield	0	100.00%	0	100.00%	0	100.00%	0	100.00%
Norman FireBreak	0	100.00%	0	100.00%	180	91.24%	5	99.48%
Sophos Anti-Virus	0	100.00%	8	99.80%	0	100.00%	15	99.33%
VirusBuster VirusBuster	0	100.00%	0	100.00%	124	92.59%	21	99.45%

Once an on-demand scan was started, the details were displayed in another window, while the product chugged confidently through the test set. Although a fair smattering of zoo viruses were missed, nothing in the ItW test set went undetected and the product generated no false positives. As a result, *Norman* also wins a VB 100% award.

Sophos Anti-Virus 4.07.0

ItW Overall	100.00%	**Macro**	99.80%
ItW Overall (o/a)	100.00%	**Macro (o/a)**	99.80%
Standard	99.33%	**Polymorphic**	100.00%

Sophos has done away with its old single-self-extracting-NLM style, and the product now provides a collection of NLMs and data files, much like most of the other products. Once copied to the server and run, the program creates all the folders it needs, demanding a user ID to 'integrate into NDS'. Updating was achieved by dropping identity files into the appropriate folder and reloading, but an automated system is available, administered by a *Windows* console.

The single-screen GUI is fairly straightforward and informative, with a menu top left and the rest of the screen showing stats and figures. One small annoyance was that the path to be scanned could not be edited once entered, and had to be deleted and replaced; this made running separate scans of several folders with the same root path rather frustrating. Another was the truncating of filenames in the log. These minor issues aside, *SAV* detected everything in the wild, threw no false positives, and did very well for speed; a VB 100% award for its performance.

VirusBuster VirusBuster 2006 for NetWare Servers v2.03.006-4.03.012

ItW Overall	100.00%	**Macro**	100.00%
ItW Overall (o/a)	100.00%	**Macro (o/a)**	100.00%
Standard	99.45%	**Polymorphic**	92.59%

VirusBuster, with its handful of NLMs and folder of data files dropped into a folder under SYS:/SYSTEM and added to the search path, demanded a licence key before activating, and then presented me with another uncluttered screen – just a small menu in the centre, surrounded by a sea of blue stripes. I found the controls a little unintuitive at first, with paths for scanning entered under 'Domain management' and scans of these paths initiated from 'Runtime options', but once this was figured out everything seemed to work reasonably well.

This was the only product to cause one of my servers to 'abend' (which was a big surprise to me – in my previous *NetWare* experience this happened fairly regularly). It occurred during some rather cavalier starting and stopping of scans of an entire SYS volume, but despite a few attempts I couldn't get it to reproduce the feat. During the clean set scanning, it also snagged on a file and had to be unloaded quite forcibly. Being in a patient and forgiving

Hard Disk Scan Rate	Executables			OLE Files		Zipped Executables		Zipped OLE Files		Dynamic files	
	Time (s)	Throughput (kB/s)	FPs [susp]	Time (s)	Throughput (kB/s)	Time (s)	Throughput (kB/s)	Time (s)	Throughput (kB/s)	Time (s)	Throughput (kB/s)
CA eTrust (InoculateIT)	222.0	2899.3		12.0	6611.1	122.0	1306.7	27.0	2763.2	30.0	1608.1
CA eTrust (Vet)	248.0	2595.4		13.0	6102.6	110.0	1449.2	29.0	2572.7	23.0	2097.5
Doctor Web Dr.Web	319.0	2017.7	1	19.0	4175.5	106.0	1503.9	19.0	3926.7	35.0	1378.4
ESET NOD32	77.0	8359.1		8.0	9916.7	48.0	3321.2	14.0	5329.1	12.0	4020.2
Kaspersky Anti-Virus	313.0	2056.4		22.0	3606.1	116.0	1374.3	24.0	3108.6	56.0	861.5
McAfee NetShield	433.0	1486.5		50.0	1586.7	257.0	620.3	59.0	1264.5	54.0	893.4
Norman FireBreak	195.0	3300.8		14.0	5666.7	34.0	4688.7	15.0	4973.8	174.0	277.3
Sophos Anti-Virus	164.0	3924.7		14.0	5666.7	53.0	3007.9	14.0	5329.1	23.0	2097.5
VirusBuster VirusBuster	394.0	1633.6	[1]	17.0	4666.7	409.0	389.8	82.0	909.8	78.0	618.5

mood during my first comparative, however, I managed to coax it gently through the rest of the tests. Detection of infected files was solid, with 100% of the ItW samples found, and labelling a single clean set file 'suspicious' was not enough to deny *VirusBuster* its VB 100% award.

CONCLUSIONS

Perhaps thanks to using a combination of the very latest version of the OS and some fairly standard hardware, I experienced few of the problems with *NetWare* that made it the bane of my predecessor's life. Likewise the products, despite a few minor irritants such as the unstoppable sending of *NetWare* alert popups to clients during on-access testing (and the associated incessant beeping), caused few headaches once I came to understand their layout.

I was struck, as Matt has been in previous reviews, by the ever-widening split between the group of products endeavouring to provide an up-to-date, user-friendly experience and those sticking with their tried-and-trusted, simple console interfaces (or, in the case of *NOD32*, the command line). *NetWare* itself reflects this dichotomy, with much of its administration yanked out of the hands of the pared-down console tools and replaced with ConsoleOne snap-ins and web management systems, to the chagrin of many veteran admins and the delight of others.

One interesting anomaly was the contrast in scan rates, and lack of contrast in detection, between the most pared-down and the most idiot-proof products. *McAfee*'s client console is clearly designed to be usable by anyone with a bare minimum of computer skills. This was at the opposite end of the throughput test from the techie-pleasing, command-line-driven *NOD32*, which zipped through the test sets in seconds, while *NetShield* ambled slowly along, way behind the pack. The two were equal top in terms of thoroughness in detecting infections though, with both products missing nothing whatsoever across all test sets.

Detection rates were generally high all round, with developers having had several weeks to get their ItW virus definitions up to speed. With little time available to update the clean test set or expand on the zoo collection, most products' detection rates in the zoo sets had changed little since the last round of tests; nevertheless, as *Dr.Web*'s bit of bad luck shows, false positives can always creep in. It is clear that I will have to get to work improving and expanding the *VB* test sets, in order to give the products more of a run for their money in the next test.

Test environment:

Servers: Identical 1.6 GHz Intel Pentium machines with 512 MB RAM, 20 GB dual hard disks, DVD/CD-ROM and 3.5-inch floppy drive, running *Novell 'Open Enterprise Server', NetWare 6.5 Support Pack Revision 5, Server version 5.70.05*.

Clients: 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 *Novell NetWare Client version 4.91.2.20051209* installed on *Windows XP Professional SP2*.

Virus test sets: Complete listings of the test sets used are at http://www.virusbtn.com/Comparatives/NetWare/2006/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

ECCE2006 will be held 12–14 September 2006 in Nottingham, UK. This will be the second E-Crime and Computer Evidence Conference to be held in Europe. For full details, including a call for papers, see http://www.ecce-conference.com/.

The Gartner IT Security Summit 2006 takes place 18–19 September 2006 in London, UK. For full details see http://europe.gartner.com/security/.

ISACA's eighth annual Network Security Conference takes place 18–20 September 2006 in Las Vegas, NV, USA. The conference will offer 90-minute and half-day sessions on a range of security topics including: physical security issues, web security environment, application security, hacking concepts and tools, encryption concepts and techniques, intrusion detection and prevention systems, wireless network security, database security and continuous security monitoring. For details see http://www.isaca.org/.

HITBSecConf2006 will take place 18–21 September 2006 in Kuala Lumpur. Seven tracks of hands-on technical training sessions run on 18 and 19 September, followed by a two-stream conference on 20 and 21 September. Full details of the training sessions and conference programme, as well as online registration, can be found at http://www.hackinthebox.org/.

T2'06 will be held 28–29 September 2006 in Helsinki, Finland. The conference focuses on newly emerging information security research. All presentations will be technically oriented, practical and include demonstrations. See http://www.t2.fi/uutisia.en.html.

COSAC 2006, the 13th International Computer Security Symposium, takes place 1–5 October 2006 in County Kildare, Ireland. The COSAC Forum gives attendees the chance to address topics of immediate and direct relevance to their organizations and get feedback and reality-based suggestions from other practitioners facing the same types of issues, albeit in different industries or stages of evolution or political turmoil in their security programs. For details of this fully residential event see http://www.cosac.net/.

The SecureLondon Workshop will be held on 3 October 2006 in London, UK. For details see https://www.isc2.org/cgi-bin/isc2event_information.cgi.

Mobile Security takes place 3–5 October 2006 in London, UK. The conference will include 12 operator case studies and a pre-conference workshop entitled 'Effectively securing premium content through interoperable DRM'. For more information see http://www.informatm.com/security/?src=vbn.

Black Hat Japan 2006 takes place 5–6 October 2006 in Tokyo, Japan. Unlike other Black Hat events, Black Hat Japan features Briefings only. For more information see http://www.blackhat.com/.

The 16th Virus Bulletin International Conference, VB2006, will take place 11–13 October 2006 in Montréal, Canada. Email vb2006@virusbtn.com for details of sponsorship opportunities. Register online at http://www.virusbtn.com/.

RSA Conference Europe 2006 takes place 23–25 October 2006 in Nice, France. Online registration and full details of the conference agenda are available now at http://2006.rsaconference.com/europe/.

Infosecurity USA will be held 24–25 October 2006 in New York, NY, USA. See http://www.infosecurityevent.com/.

AVAR 2006 will be held 4–5 December 2006 in Auckland, New Zealand. See http://www.aavar.org/.

The International Conference on Human Aspects of Information Security & Assurance will be held 10–12 July 2007 in Plymouth, UK. The conference will focus on information security issues that relate to people – the methods that inform and guide users' understanding of security and the technologies that can benefit and support them in achieving protection. For more details, including a call for papers, see http://www.haisa.org/.

SUBSCRIPTION RATES

Subscription price for 1 year (12 issues):

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

Corporate rates include a licence for intranet publication.

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

Editorial enquiries, subscription enquiries, orders and payments:

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

Tel: +44 (0)1235 555139 Fax: +44 (0)1235 531889

Email: editorial@virusbtn.com Web: http://www.virusbtn.com/

vb Spam supplement

CONTENTS

NEWS & EVENTS

MARKET CONSOLIDATION

Anti-spam and messaging security company *CipherTrust* announced last month that it is to be acquired by *Secure Computing*.

CipherTrust – which currently has an approximately 20% market share in the messaging security market – will be purchased by *Secure Computing* for a total of $273.6m in cash and stock. At the close of the deal *CipherTrust* shareholders will own 14% of *Secure Computing*. The purchase agreement will enable *Secure Computing* to add *CipherTrust*'s messaging security appliances to its range of security products. Combined assets of the merged companies are estimated to exceed 18,000 enterprise customers with 1,700 resellers in 106 countries.

The deal is expected to close in early September.

NEW AND REVISED LAWS

One of the world's top spam-sending hot spots is currently preparing for the launch of its first anti-spam laws. Together with the rest of China, the Chinese territory of Hong Kong has been named in a number of recent reports (e.g. *Sophos*, *Commtouch* and *Spamhaus*) as the world's second-most prolific spam-sending region. However, it is hoped that the new laws – which promise fines and prison sentences for those who fall foul of the rules – will help bring about a change in the mailing habits of the territory.

Although the full text of the proposed Unsolicited Electronic Messages Bill has yet to be revealed by the Hong Kong government, much of its content has been described by officials.

Companies that are based or trade in Hong Kong will be penalised if they engage spammers to market their products. The law may also allow action against individuals who authorise spam campaigns if they are in Hong Kong at the time the spam is sent.

According to Joseph Wong, Hong Kong's secretary for commerce, industry and technology, 'The law covers all electronic messages with a Hong Kong connection. If it originated from Hong Kong, or is sent to Hong Kong, it is within the ambit of the bill.'

A range of penalties is proposed, with those who route spam through open relays, or hack into other computers to relay spam facing the harshest penalties. These offences may result in fines of up to US$130,000 and prison sentences of up to five years.

The bill, which covers telephone, fax and instant messaging as well as email, is expected to be passed into law later this year.

Meanwhile, the UK government has admitted that it may be forced to revisit its heavily criticised anti-spam legislation.

The Privacy and Electronic Communications Regulations, which was introduced in December 2003, was criticised for the fact that, although it bans the sending of spam to individuals, it does not stop spammers targeting businesses.

Furthermore, data protection watchdog the Information Commissioner's Office (ICO) has said that the legislation does not provide sufficient powers to track down and prosecute spammers – and leaves it powerless against those that originate from outside the UK. Under the current legislation the ICO can only take enforcement action against spammers based in the UK, and the maximum fine that can be imposed in the magistrates courts is £5,000.

Recently, however, trade and industry minister Margaret Hodge revealed that the government is thinking of revisiting the legislation. She told MPs: 'The government is actively considering whether to revise the relevant legislation. DTI officials have a continuing dialogue with Internet service providers regarding steps that can be taken to reduce spam. We also continue our efforts to achieve greater international co-operation.'

SUPPLY OF DATA TO SPAMMERS STOPPED

A US man has been charged with stealing a database of US physicians with the intention of selling it on to spammers.

Forty-six-year-old William Bailey, Jr is alleged to have downloaded the contact details of 80,000 members of a database maintained by the American College of Physicians (ACP) with the intention of selling the details on his own website. Bailey's website dr-411.com advertises professional organization member databases for sale, including addresses and email addresses for doctors, dentists, lawyers and estate agents. Bailey faces a maximum penalty of 55 years in jail and $2.75m in fines if found guilty.

ANTI-PHISHING BEST PRACTICES

A new set of best practices to combat phishing has been released by the Anti-Phishing Working Group (APWG) and Messaging Anti-Abuse Group (MAAWG), to help ISPs and mailbox providers better police their infrastructures and filter the traffic traversing their networks.

The two industry groups joined forces to develop the guidelines, which include:

- Two-way filtering of traffic to prevent phishing emails from reaching consumers and to alert ISPs and mailbox providers when their own servers are being used for sending phishing emails.
- The use of IP blacklists to close down temporarily servers that have been co-opted for phishing attacks; the use of URL-based filters to help ISPs filter outbound customer traffic to known phishing IP addresses, domains or URLs.
- Filtering or rejecting email if it can unequivocally be determined to be forged; disabling images and hyperlinks in email from untrusted sources.
- Blocking access to known phishing websites during attacks.

The recommendations also highlight the importance of educating consumers to check for website certificate authenticity before submitting personal information, to report scams to the Federal Trade Commission or equivalent anti-fraud organizations, and alerting financial institutions when they are the target of phishing campaigns.

'Anti-Phishing Best Practices for ISPs and Mailbox Providers' can be downloaded from http://antiphishing.org/reports/bestpracticesforisps.pdf.

EVENTS

The Text Retrieval Conference (TREC) 2006 will be held 14–17 November 2006 at NIST in Gaithersburg, MD, USA. More details, including information on how to participate in the TREC 2006 Spam Track, can be found at: http://plg.uwaterloo.ca/~gvcormac/spam/.

FEATURE

SPUTR: A PROPOSAL FOR THE UNIFORM NAMING OF SPAMMER AND PHISHER CONTENT TRICKS

John Graham-Cumming
Independent consultant, France

I have been tracking the tricks used by spammers in the bodies of their messages since January 2003. Three years on, I have collected 55 distinct tricks and published them on *The Spammers' Compendium* website [1]. When I first started publishing the site I gave each of the tricks a humorous name (such as 'Camouflage' or 'Honey, I shrunk the font'), and some of these names have entered popular use (such as 'Hypertextus Interruptus', which is enshrined in the *SpamAssassin* test INTERRUPTUS).

TRICKS IN THE WILD

The trick count has been growing steadily over the last three years: Figure 1 shows the number of tricks in *The Spammers' Compendium* by calendar quarter. It is interesting to note that trick innovation or discovery seems to slow down in the fourth quarter of each year – perhaps indicating that spammers are in the middle of spamming their Christmas campaigns at that time, and not spending time on modifying their software.

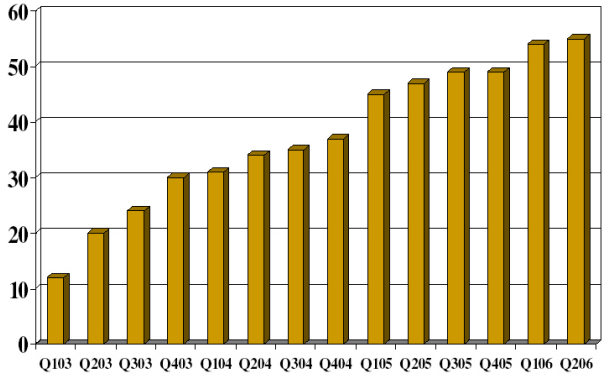

Figure 1: Trick count by calendar quarter.

Entries are made in *The Spammers' Compendium* when the tricks have been identified by me in spam seen in the wild in my spam traps, or in spam emailed to me by volunteers.

Submitters receive credit in *The Spammers' Compendium* for submitting a new trick.

While the humorous names make good copy for journalists writing about the latest devious spammer trickery, they are less useful to people working in anti-spam research because they do not, in themselves, convey much information. In this article (and the related blog post [2]) I propose a drier, but more information-rich, naming scheme that can be used to refer to spammer and phisher content tricks.

TIME FOR A NAMING SCHEME

At the 2004 Virus Bulletin conference I presented a paper (see [3]) in which I analysed some trends in the use of spammers' tricks by examining the appearance of various tricks (as extracted from *The Spammers' Compendium*) against a large corpus of spam supplied by *Sophos*. One of the problems in that analysis was that I was forced to write code to identify the tricks in *The Spammers' Compendium* and I also had to explain each trick as the names conveyed little information.

To remedy that situation and provide a foundation on which other authors and vendors can build research into spammer trickery I think it's time for a uniform naming scheme for these tricks.

In the uniform naming scheme, which I am calling the *Spam/Phish Uniform Trick Repository*, or *SPUTR*, each name consists of three '!'-separated parts: a purpose, a name, and a technology. The purpose is the reason for the trick (for example, the trick is used to obscure a URL, or to insert innocent words). The name is derived from the current *Spammers' Compendium* pejorative name. The technology identifies the way in which the trick is coded (for example, with HTML or MIME).

Table 1 contains a list of proposed 'purposes' that can be used to categorize tricks.

For a single name there could be multiple tricks using different technologies (e.g. some tricks might be implemented using HTML or CSS), or tricks intended for different purposes (words might be inserted to fool a Bayesian filter or break a hash).

Table 2 shows the 'technologies' that would be recognized in the naming scheme:

BWO	Bad word obfuscation	Making it hard for a filter to parse potentially bad words (e.g. Viagra).
GW	Good word insertion	Adding words likely to confuse a statistical filter.
HB	Hash busting	Inserting randomness designed to make message hashing hard.
TA	Tokenization avoidance	Preventing a filter from tokenizing a message.
UH	URL hiding	Hiding a URL so that a user is fooled into clicking an incorrect link.
UO	URL obfuscation	Making it hard for a filter to identify a URL and check it against a black list.
WB	Web bugs	Inserting a beacon that tells the spammer that a message has been read.

Table 1: Trick purposes.

CSS	Use of CSS
HTML	Any HTML without using CSS
Javascript	Use of Javascript for trickery
MIME	Manipulation of MIME
Plain	Plain text

Table 2: Technology identifiers.

For example, the original Invisible Ink trick, written using HTML, would be referred to as:

GWI!Invisible!HTML

while a CSS variant would be:

GWI!Invisible!CSS

Names would be generated only for tricks that have been seen in the wild.

With such uniform naming it would be possible to analyse spams and phishes (perhaps even specific recognizers for each trick could be written) and the trends built up over time to see how individual tricks and individual classes of tricks are changing.

Table 3 shows the proposed mapping from the current *Spammers' Compendium* names to the SPUTR name.

The Big Picture	TA!BigPicture!HTML
Invisible Ink	GWI!Invisible!HTML and GWI!Invisible!CSS
The Daily News	GWI!BigTag!HTML
Hypertextus Interruptus	BWO!Interruptus!HTML
Slice and Dice	TA!SliceNDice!HTML
MIME is Money	GWI!PlainNotHTML!MIME
Lost in Space	BWO!Space!Plain
Enigma	UO!Enigma!HTML
Script Writer	TA!Script!Javascript
Ze Foreign Accent	BWO!Accent!Plain
Speaking in Tongues	HB!Tongues!Plain

The Black Hole	BWO!BlackHole!HTML
A Numbers Game	BWO!Numbers!HTML
Bogus Login	UO!BogusLogin!HTML
Honey, I Shrunk the Font	GWI!ShrunkFont!HTML
No Whitespace, No Cry	TA!NoWhitespace!Plain
Honorary Title	GWI!Title!HTML
Camouflage	GWI!Camouflage!HTML
And in the right corner	HB!RightCorner!Plain
A Form of Desperation	GWI!Form!HTML and BWO!Form!HTML
It's Mini Marquee!	GWI!Marquee!HTML
You've been framed	BWO!Framed!HTML
Control Freak	TA!ControlFreak!Plain
Don't Cramp My Style	GWI!Style!CSS
The Microdot	BWO!Microdot!CSS
WYSI_not_WYG	UH!WYSINotWYG!Javascript
Ultra	See Engima
Internet Exploiter	UH!InternetExploiter!HTML
Style Wars: Episode 1	Included in other tricks
The tURLing Test	UO!TurlingTest!Plain
Flex Hex	BWO!FlexHex!CSS
Sound of Silence	WB!Silence!HTML
Blankety Blank	BWO!BlanketyBlank!HTML
Doing the Splits	BWO!Splits!Plain
But is it art?	BWO!ASCIIArt!Plain
Absolute Zero	Same as Control Freak
Spell Breaker	BWO!Splelnig!Plain
About Face	BWO!AboutFace!HTML
Catch a Wave	TA!Wave!HTML
Treasure Map	UH!TreasureMap!HTML
You cannot be serious	UO!Mcenroe!HTML
The Matrix	TA!Matrix!Plain
Sticky Fingers	BWO!StickyFingers!Plain
Floatation Device	TA!Floatation!CSS
The Small Picture	TA!SmallPicture!HTML
Chop	GUI TA!ChopGUI!HTML/ HB!ChopGUI!HTML
Big Header-ed	?
The Rake	BWO!TheRake!CSS
Now you see it; now you don't	BWO!Copperfield!CSS
Slick Click Trick	UH!Caption!HTML
Whiter shade of Pale	TA!Pale!HTML

Table 3: Trick name mapping.

COOPERATION

If the anti-spam and anti-phish community gets together now it may be able to avoid the mess that exists in the anti-virus industry where vendors compete to release information about viruses and each have their own way of naming them.

Worse, the current unifying malware scheme maintained by MITRE (the Common Malware Enumeration or CME; see http://cme.mitre.org/) unifies virus names by providing a simple identifier for each that contains absolutely no information. For example, the Kukudro.C worm is currently assigned the uninformative name 'CME136'.

In order to help the anti-spam and anti-phish community I propose to:

1. Maintain a website containing the uniform naming scheme and keep it updated as new spammer tricks are reported to me;
2. Allow any organization to use the names freely and identify themselves as a user by including their name or logo on an appropriate page on the site without any form of compensation;
3. Accept reports of new spammer and phisher trickery for inclusion on the website;
4. Host a mailing list for all interested parties so that tricks can be discussed and named;
5. Manage an open source project that creates software that can analyse an RFC822 message and output the tricks used.

In order to do that I would like the support of at least five major email security companies in the form of a decision to use the SPUTR names in their own research and publications.

Undoubtedly there will be many things about this proposal that old anti-virus hands, and those fighting email security problems would like to modify or comment on; please send your comments to jgc@jgc.org.

REFERENCES

[1] The Spammers' Compendium. http://www.jgc.org/tsc/.

[2] Graham-Cumming J. Proposed uniform naming scheme for spammer/phisher content trickery. http://www.jgc.org/blog/2006/06/proposed-uniform-naming-scheme-for.html.

[3] Graham-Cumming J. *The Waxing and Waning of* Spammers' Trickery. Proceedings of the Virus Bulletin International Conference, 2004. http://www.virusbtn.com/conference/vb2004/abstracts/jgrahamcumming.xml.