ISSN 0956-9979 MARCH 2001

VIRUS BULLETIN

THE INTERNATIONAL PUBLICATION ON COMPUTER VIRUS PREVENTION, RECOGNITION AND REMOVAL

IN THIS ISSUE:

• **Zmists of time:** Peters Ferrie and Ször from *Symantec* tackle the extremely complex binary virus W95/Zmist while *VirusBuster's* Gabor Szappanos ponders the implications of Davinia. Virus Analyses start on p.6.

• **Expert opinions:** AV specialists share their views on the problems associated with the release of Mac *Office 2001* and debate the potential of PHP viruses, starting on p.10.

• **Corporate kudos:** from *CERTCC* in Korea to Boston's *Fidelity Investments*, our Features this month track the challenges facing corporate networks around the world, from p.14.

CONTENTS

COMMENT

NEWS

VIRUS ANALYSES

OPINIONS

FEATURES

PRODUCT REVIEWS

VIRUS BULLETIN **©2001 Virus Bulletin Ltd, The Pentagon, Abingdon, Oxfordshire, OX14 3YP, England.**
www.virusbtn.com /2001/$0.00+2.50

COMMENT

“ … an important process needed here is ‘due diligence’ ”

Growing Pains

At VB’98 in Munich the closing panel discussion made many predictions, as it always does. One was what the future AV landscape would look like from a ‘player’ perspective. Most on the panel thought there would be a ‘thinning out’ of the companies in the AV field and that there would be only about five players left globally. At the time, there were about 30 AV companies worldwide. Four led the pack in market share, and of those four, three were software companies involved in various software ventures. Approximately three years later, the landscape looks pretty much the same. The market is still growing; there are a few new players and some of those around in ’98 have fallen by the wayside.

In these growing times, there still seem to be growing pains. Some of these include naming conventions – most customers’ number one pet peeve is who detects what first and when. Others involve taking a shot at the guys on top, and at the bottom for that matter, in order to make those quarterly numbers. One of the most visible growing pains seems to be the on-going problem of obtaining samples from companies who want a by-line, but who can’t get out of their own way to create a process whereby they can achieve that and stay out of the dog house at the same time.

Over the past couple of months or so we’ve seen two separate occasions where a virus threat has been profiled and a problem has arisen as a result of two AV companies – *Trend Micro* and *Panda* – and their failure, in my opinion, to learn to grow as a business before they learned to fly. Due to misunderstandings about how to participate in the profiling process, these companies created single points of failure and doing so caused angst for many others along the line. Those directly affected were the customers. Other AV companies were indirectly affected but took the direct heat from the customers.

This problem was supposed to be circumvented some time back. A group was formed to make sure those ‘BIG’ companies didn’t roll over the little guys and everyone got what was needed. Funny how the *really* big guys were out in the cold on these last two occasions, and really haven’t been involved in a problem like this for some time.

For a company or an entity to succeed it must follow certain practices, most of which have been around for some time. Organization is probably the key word in this formula, but an important process needed here is ‘due diligence’. Researchers exercise this when deciding if they will exchange samples with another researcher or person. If they don’t, and get burned, they are the ones who suffer the consequences. We saw both the companies involved suffer some consequences in this latest situation.

Due diligence needs to be carried out as this growth continues in all areas. It must permeate upwards from researcher to researcher, researcher to business unit and from company to company. This is a must if you are going to choose to get involved at that level, which from what I can see didn’t appear to happen this time as things got revved up. I’m curious to know what has changed in the processes since this latest round of ‘whodunnit’. I’m sure the cards and love letters will come pouring in, and the growing pains will continue.

At the end of the day, AV is a service-based business, but one that is unique in that there is reliance on one another in a profile situation. It doesn’t matter if you’re big or small, in most cases when you say ‘boo!’ everyone listens and runs to the Web to get what they need. For this industry to alleviate some of its growing pains, a little discovery from the business side of things wouldn’t hurt. Research and business (subliminal message; due diligence), research and business, say it with me – ‘due diligence’. Make sense? Keep repeating it, at some point the customers will appreciate that we get it, and from there harmony and peace will reign and the AV world as we know it will live happily ever after.

Vincent Gullotto, McAfee AVERT Labs, USA

NEWS

Tennis Elbow

Yet again it appears that the sole criterion for the issuing of AV vendor press releases is that it has to be a day of the week with a ‘y’ in it. Official festivals and holidays count for double points – or so the slew of laughably vague Valentine’s Day premonitions implied. It amuses us at *Virus Bulletin* that here we have the self-styled protectors of the information superhighway reduced to issuing ‘official’ advice on last minute card shopping for ‘bashful well-wishers’, musing on ‘the risk attached to office romances’ and even mourning their customers as a pack of ‘unromantic geeks’. Ouch!

When another fairly trivial, if widespread, email worm obligingly came along, the marketing departments went into ecstasies of excitement. Rapturous AV vendors dressed the AnnaKournikova.jpg.vbs worm in such emotive terms as able to cause ‘email storms’ at ‘record breaking speeds’, not to mention adding to the confusion by throwing a wide selection of names at this ‘new’ piece of malware.

Accordingly, the *VB* ‘Thank Goodness’ award for the most sensible and comprehensive press release is awarded jointly to *Norman* and *GeCAD*. Special mentions go to *Sophos* for managing to get the word ‘groin’ published in an official release so close to Valentine’s Day, and *NAI* for its elaborate disclaimer should the tennis star find herself in a particularly litigious mood. *F-Secure* didn’t do badly either, receiving the ‘Jolly Good Sports’ award for capitalising on their ‘Swedish partners’ *Atremo AB’s* part in tracking down the worm’s author.

Joking apart, there is a serious side to all this ‘publicity’. We were astonished to see *Kaspersky Lab* handwashing a large red rag on behalf of adolescent bulls everywhere in publishing detailed information on the very kit used to write the worm. And the *Aladdin Knowledge Systems* marketing department must be due for a reshuffle. *AKS* issued a highly suspicious ‘Valentine’s Day virus warning’ virtually none of which could be substantiated – the dubious sample sent to REVS didn’t work, it doesn’t ‘autorun’ from an HTML email (it posts itself as a VBS attachment) and it wasn’t clever to name a lame LoveLetter variant after its author. Nevertheless, *Panda* also saw fit to post a warning on its Web page. Enough already! ▮

Just like that! PIF!

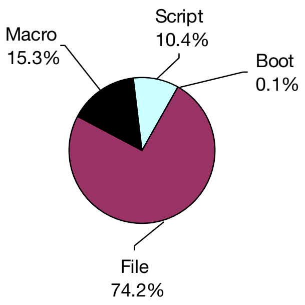
An early February edition of *IT Week* reported at least one company experiencing problems with *Trend Micro’s OfficeScan* failing to detect MTX and other viruses with .PIF extensions in default mode. *Trend* declined to submit for our *Windows ME* Comparative, and the problem has since been fixed, but it’s always gratifying when our rigorous and controversial testing protocol is vindicated ▮

Prevalence Table – January 2001

Virus	Type	Incidents	Reports
Win32/Navidad	File	1912	31.0%
Win32/Hybris	File	1336	21.6%
Win32/MTX	File	1034	16.7%
LoveLetter	Script	320	5.2%
Kak	Script	299	4.8%
Ethan	Macro	276	4.5%
Divi	Macro	114	1.8%
Win32/Prolin	File	100	1.6%
Onex	Macro	90	1.5%
Myna	Macro	74	1.2%
Laroux	Macro	71	1.1%
Manalo	Macro	68	1.1%
Tristate	Macro	51	0.8%
Marker	Macro	40	0.6%
Win32/Ska	File	35	0.6%
Win32/Funlove	File	30	0.5%
Thus	Macro	27	0.4%
Win32/Pretty	File	27	0.4%
Win32/Msinit	File	22	0.4%
Win32/QAZ	File	22	0.4%
Class	Macro	18	0.3%
Cap	Macro	13	0.2%
Stages	Script	13	0.2%
Win95/CIH	File	13	0.2%
Jini	Macro	12	0.2%
Netlog	Script	9	0.1%
Others		151	2.6%
Total[1]		**6177**	**100%**

[1] The Prevalence Table includes a total of 151 reports across 45 further viruses. Readers are reminded that a complete listing is posted at http://www.virusbtn.com/Prevalence/.

Distribution of virus types in reports



LETTERS

Dear Virus Bulletin

Gone with the Wild

Every day, I see more and more viruses which depend on some kind of data downloaded from the Internet for their evolution and progress. For example, several common viruses download Trojans from the Internet and install them in the system. Alternatively, some viruses are simply unable to replicate without certain files located in special places on the Internet.

Let's take for example the infamous Davinia virus, which was reported in the wild by Spanish AV vendor *Panda Software*. This virus cannot replicate without a copy of itself being available at a specific Internet location on the Spanish Web site 'terra.es'. If this virus were to have been reported by two different WildList reporters then it would have got into the WildList.

Moreover, even if the Web page were no longer available and the virus unworkable, it would still stay in the WildList for at least six months when, if no subsequent reports were received, it would be removed automatically. I don't think this is correct from a user's point of view. If the virus doesn't work any more, then it can't be ItW and thus, it should not be included in the WildList.

Another case is the known virus JS/Unicle. This one downloads a couple of Trojans from a specific Internet location, and runs them. However, they are no longer available for download, as the respective Web page was removed, so the relevant Trojans have no chance of being found any more on users' machines. From this point of view, the WildCore honestly provides only the JS/Unicle sample itself, without including the Trojans downloaded by the virus.

However, I believe that some anti-virus testing institutions include the respective Trojans in their tests, tests which are supposedly based on the latest WildList. I personally don't find that fair – the specific Trojans downloaded by JS/Unicle are something totally unrelated (now) to the virus itself, and should not be used while testing anti-virus programs for detection of the latest WildList.

Therefore, I think it would be totally unfair to punish a product for not detecting some malware which simply cannot technically be found ItW any more, in a test which reports if the product is able to detect all the malware which is currently found ItW.

Don't get me wrong – I am not saying that things which are not in the wild any more should not be detected by anti-virus products. They definitely should be detected, but a product should not be punished for not detecting them in a specific test which verifies the ability of a product to detect malware which is *currently* in the wild.

Costin Raiu
Kaspersky Lab
Romania

… VB Responds

Virus Bulletin broadly agrees with Costin's view about virus-related Trojans in testing. *VB* is currently in discussion with other anti-virus groups and testers about this issue. A more definitive statement will be made in the next Comparative Review in the April issue.

However, we should like to make it clear that the example Costin uses of the JS/Unicle EXE file was only ever included in our Standard test-set, never in our ItW set. This, we felt, could be included as representing a Trojan component of a virus. Thus far, we have never included Trojans, associated with viruses or otherwise, in any of our In the Wild test-sets.

Matt Ham
Virus Bulletin
UK

Review Request

This letter is a request to *Virus Bulletin* and the other anti-virus product reviewers to make an inevitable change to current practice now, rather than later. Accordingly, a copy has been sent to the *ICSA*, *Secure Computing*, VTC Hamburg, University of Magdeburg, University of Tampere and also to Joe Wells of the *WildList Organization* even though they would probably see it here.

Very simply, the change I want to see is this: please exclude all 'Old Fashioned DOS File Viruses' (OFFVs) from test-sets, as soon as possible. The rationale is simple. No one is interested in them any more! If you look at the magazines, there is almost no mention of them in the articles, or the lists of current viruses. However, a lot of them are in the test suites, because they've never been taken out.

Many anti-virus vendors have long ago taken the decision only to process OFFVs, if they come in from the field. The effect is that the 12 virus collections I receive each month, now contain very few. I expect they will reduce to zero within two years. If you accede to my request, there will be three major effects, and a fourth fairly minor one.

- You may need to enhance your test suites with more macro and *Windows* viruses, and perhaps review more Trojans.
- Instead of everybody achieving detection rates in the high nineties, the range will be increased.

- You may get some ‘aggro’ from the vendors whose detection rates drop sharply, and you will have to be prepared to provide lists and samples of what has been missed.
- You will help to speed up the death knell for the OFFVs.

There is one argument against my proposal, namely that Far Eastern users are still affected, and OFFVs are still being written there. However, if that were a valid argument, the Far Eastern users would need more than the top-up figures which dilute the review comparisons, and they’re not getting any more!

If the above reviewers agree, they should probably discuss it between themselves, and agree a common start date, because that will spread the ‘aggro’ resulting from the occasional poor product, rather than allow it to home in on the first reviewer to make the change. I look forward to your comments, and particularly to agreement as to when the change can be made.

Peter Morley
McAfee AVERT
UK

Blast from the Past

As I was analysing W97M/Serlock.B I found something interesting which took me back five years to when I was learning anti-debugger virus tricks. This virus looks simple, except that the 14 lines in the code that actually copy the virus code from one document to another are encrypted and REM’d out. The structure of the virus is the following:

```
Sub Document_Open
if code_is_encrypted then
 decrypt_code
end if
End Sub
Sub Document_Close
if code_is_decrypted then
 encrypt_code
end if
encrypted_propagation_code
End Sub
```

Whenever a document is opened (and the normal template is already infected) the virus decrypts the propagation code in the Document_Close macro, which makes the propagation code available. When the uninfected document is about to be closed, the virus encrypts the propagation code, then copies the encrypted code into the document.

I was surprised to see that it works. The Document_Close procedure first encrypts and REMs out the propagation code, yet the code will be executed (as if it were still available). More interestingly, the encrypted code version will be copied to the infected document. It seems that when the Document_Close procedure is executed, the whole procedure code is fetched to memory for execution. When the encryptor is executed, it will encrypt the code stored in the document macro storage, but VBA does not care to synchronize the already loaded macro code, and carries on with the execution. The code copies the macro code from the macro storage (which is already modified and encrypted) to the uninfected document. This prefetching works on procedure level as opposed to module level, otherwise the Document_Open procedure wouldn’t be able to decrypt the propagation code and the virus wouldn’t work at all. All this somewhat reminded me of the tricks that DOS viruses played with the processor prefetch queue of the *Intel 486* processors.

Gabor Szappanos
VirusBuster
Hungary

Media Responsibility

Media excitement over the outbreak of the prosaically named VBS/VBSWG.J virus – more commonly dubbed Anna Kournikova by journalists – rapidly gave way to interest in the writer of this latest ‘Internet killer’. This recent worm was ‘written’ by someone using the handle OnTheFly. In real life this is probably the 20 year-old Dutchman Jan de Wit who, perhaps appropriately, hails from a town called Sneek.

All good column-inch filling stuff, but what concerns me is that many reputedly ‘responsible’ media outlets went a step too far in their detailed reporting of this. The intrusion into Jan de Wit’s life is unfortunate (particularly if the journalists who put the clues together here fingered the wrong guy) but that is not my concern.

Most media outlets latched onto OnTheFly/de Wit’s admission ‘I don’t know any programming languages’ and his use of a ‘Visual Basic Worm Generator’. Reporting this fact is, on balance, a good thing. It raises awareness that there are reprobates prepared to make life easy for wannabe digital miscreants who are too untalented to adopt that role unaided. What prompted me to write this letter was the level of detail with which this was reported. Some outlets reported the exact name of the generator kit and a few provided hotlinks from their news stories to a site carrying the kit.

Such actions are irresponsible at best, as they provide more detail than is necessary to make the important point. Those more detailed reports direct the truly untalented, and those too lazy to run a Web search, straight to a tool designed to make it easy for such stupid and naïve people to ‘create’ trouble. I implore *VB’s* readership to keep a watch for similar acts of irresponsible reporting in future and politely point out the ills of such acts to the journalists concerned and their editors, as they are often surprisingly obliging in removing such material if they receive complaints.

Nick FitzGerald
Computer Virus Consulting Ltd
New Zealand

VIRUS ANALYSIS 1

Zmist Opportunities

Peter Ferrie & Péter Ször
SARC, USA

At VB2000 in Florida, *IBM's* Dave Chess and Steve White demonstrated their research findings on 'Undetectable Computer Viruses'. Early this year, the Russian virus writer Zombie released his 'Total Zombification' magazine complete with a set of articles and viruses of his own. Ominously, one of the articles in the magazine was titled 'Undetectable Virus Technology'.

Zombie has already demonstrated his set of polymorphic and metamorphic virus-writing skills. His viruses have been distributed for years in source format and other virus writers have modified them to create new variants. Certainly this will be the case again with Zombie's latest creation – W95/Zmist.

Many of us will not have seen a virus approaching this complexity for a few years. We could easily call Zmist one of the most complex binary viruses ever written. W95/SK, One_Half, ACG, and a few others come to mind in comparison. Zmist is a little bit of everything: it is an entry point obscuring virus that is metamorphic. Moreover, the virus randomly uses an additional polymorphic decryptor.

This virus supports a unique new technique: code integration. The Mistfall engine contained in it is capable of decompiling Portable Executable files to its smallest elements, requiring 32 MB of memory. Zmist will insert itself into the code: it moves code blocks out of the way, inserts itself, regenerates code and data references, including relocation information, and rebuilds the executable. This is something never seen before in previous viruses.

Zmist occasionally inserts jump instructions after every single instruction of the code section, each of which will point to the next instruction. Amazingly, these horribly modified applications will still run as before, just like the infected executables do, from generation to generation. In fact, we did not see a single crash during the test replications. Nobody expected this to work, not even Zombie. However, it is not foolproof – it takes some time for a human to find the virus in infected files. Due to its extreme camouflage Zmist is clearly the perfect anti-heuristics virus.

Initialisation

Zmist does not alter the entry point of the host. Instead it merges itself with the existing code, becoming part of the instruction flow. However, the random location of the code means that sometimes the virus will never receive control. If the virus does run, then it will immediately launch the host as a separate process, and hide the original process (if the RegisterServiceProcess () API is supported on the current platform) until the infection routine completes. Meanwhile, the virus will begin searching for files to infect.

Direct Action Infection

After launching the host process, Zmist will check if there are at least 16 MB of physical memory installed and that it is not running in console mode. If these checks pass, then it will allocate several memory blocks, including a 32 MB area for the Mistfall workspace, permutate the virus body, and begin a recursive search for Portable Executable .EXE files. This search will take place in the *Windows* directory and all subdirectories, the directories referred to by the PATH environment variable, then all fixed or remote drives from A to Z. This is a brute force approach to spreading.

Permutation

The permutation is fairly slow because it is done only once per infection of a machine. It consists of instruction replacement, such as the reversing of branch conditions, register moves replaced by push/pop sequences, alternative opcode encoding, xor/sub and or/test interchanging, and garbage instruction generation. The same engine, Real Permutating Engine (RPME), is used in several viruses including W95/Zperm, also written by Zombie.

Infection of Portable Executable Files

A file is considered infectable if it is smaller than 448 KB, if it begins with 'MZ' (*Windows* does not support the 'ZM' form), if it is not infected already (the infection marker is 'Z' at offset 0x1C in the MZ header – this field is not generally used by *Windows* applications), and if it is a Portable Executable file. The virus will read the entire file into memory, then choose from one of three possible infection types.

There is a one in ten chance that only jump instructions will be inserted between every existing instruction (if the instruction was not a jump already), and the file will not be infected. There is the same probability that the file will be infected by an unencrypted copy of the virus; otherwise, the file will be infected by a polymorphically encrypted copy.

The infection process is protected by Structured Exception Handling which prevents crashes in the case of errors. When the rebuilding of the executable is completed, the original file is deleted and the infected file is created in its place. However, if an error occurs during the file creation, then the original file is lost and nothing will replace it.

The polymorphic decryptor consists of 'islands' of code that are integrated into random locations throughout the host code section and linked together by jumps. The decryptor integration is performed in the same way as for the virus body integration – existing instructions are moved to either side, and a block of code is placed in between them. The polymorphic decryptor uses absolute references to the data section, but the Mistfall engine will update the relocation information for these references too.

An anti-heuristic trick is used for decrypting the virus code: instead of making the section writable in order to alter its code directly, the host is required to have, as one of the first three sections, a section containing writable, initialised data. The virtual size of this section is increased by 32 KB, large enough for the decrypted body and all the variables used during decryption. This allows the virus to decrypt code directly into the data section, and transfer control to there. If such a section cannot be found, then the virus will infect the file without using encryption.

The decryptor will receive control in one of four ways: via an absolute indirect call (0xFF 0x15), a relative call (0xE8), a relative jump (0xE9), or as part of the instruction flow itself. If one of the first three methods is used, the transfer of control will usually appear soon after the entry point. In the case of the last method, though, an island of the decryptor is simply inserted into the middle of a subroutine, somewhere in the code (including before the entry point).

All used registers are preserved before decryption and restored afterwards, so the original code will behave as before. Zombie calls this last method ‘UEP’, perhaps an acronym for Unknown Entry Point, because there is no direct pointer anywhere in the file to the decryptor.

When encryption is used, the code is encrypted with ADD/SUB/XOR with a random key, and this key is altered on each iteration by ADD/SUB/XOR with a second random key. In between the decryption instructions are various garbage instructions, using a random number of registers, and a random choice of loop instruction, all produced by the Executable Trash Generator engine (ETG), also written by Zombie. It is clear that randomness features very heavily in this virus.

Code Integration

The integration algorithm requires that the host has fixups, in order to distinguish between offsets and constants. However, after infection, the fixup data are not required by the virus. Therefore, though it is tempting to look for an approximately 20 KB long gap in the fixup area, which would suggest that the virus body is located there, it would be dangerous to rely on this during scanning.

If another application (such as one of an increasing number of viruses) were to remove the fixup data, then the infection will be hidden. The algorithm also requires that the name of each section in the host is one of the following: CODE, DATA, AUTO, BSS, TLS, .bss, .tls, .CRT, .INIT, .text, .data, .rsrc, .reloc, .idata, .rdata, .edata, .debug, DGROUP. These section names are produced by the most common compilers and assemblers in use, those of *Microsoft*, *Borland*, and *Watcom*. The names are not visible in the virus code, because the strings are encrypted.

A block of memory is allocated which is equivalent to the size of the host memory image, and each section is loaded into this array at the section’s relative virtual address. The location is noted of every interesting virtual address (import and export functions, resources, fixup destinations, and the entry point), and then the instruction parsing begins. This is used in order to rebuild the executable.

When an instruction is inserted into the code, all following code and data references must be updated. Some of these references might be branch destinations, and in some cases the size of these branches will increase as a result of the modification. When this occurs, more code and data references must be updated, some of which might be branch destinations, and the cycle repeats.

Fortunately – at least from Zombie’s point of view – this regression is not infinite, so that while a significant number of changes might be required, the number is limited. The instruction parsing consists of identifying the type and length of each instruction. Flags are used to describe the types, such as instruction is an absolute offset requiring a fixup entry, or instruction is a code reference, etc. There are cases where an instruction cannot be resolved in an unambiguous manner to either code or data. In that case, Zmist will not infect the file.

After the parsing stage is completed, the mutation engine is called, which inserts the jump instructions after every instruction, or generates a decryptor and inserts the islands into the file. Then the file is rebuilt, the relocation information is updated, the offsets are recalculated, and the file checksum is restored. If there are overlay data appended to the original file, then they are copied to the new file too.

Conclusion

A few years ago several anti-virus researchers claimed that algorithmic detection had no future. We would like to take this opportunity to turn that around, by claiming that virus scanners will have no future if they do not support algorithmic detection at the database level.

It is amazing to see how polymorphic viruses become more and more advanced over the years. Such metamorphic creations will come very close to the concept of a theoretically undetectable virus. The computing environment had to change and it did change. Now, modern viruses completely support this new environment. In the next couple of years we will be able to see how complex DOS viruses would be today if the environment had not changed during the last few years.

But for the time being, we are once again one step ahead of the virus writers. ‘So, poly-encrypted permutated viral body is completely integrated with target file. Hmm … checkmate?’ Not this time, Zombie.

VIRUS ANALYSIS 2

Davinia and Goliath

Gabor Szappanos
VirusBuster, Hungary

{JS/VBS/HTML/WM97}/Davinia is just another in the seemingly endless line of email worms. While it does not display much in the way of novelty as far as coding is concerned, it certainly raises several questions worth mentioning. Davinia was written by a Spanish virus writer calling himself Onel 2. The worm was found in the wild on one Spanish Web site and allegedly caused damage in at least five companies.

Office Vulnerability

Davinia uses a relatively well-known security breach introduced by *Office 2000*. The 'Microsoft Office 2000 UA Control Scripting' vulnerability was published on 12 May 2000, after its official release in *Microsoft Security Bulletin* (MS00-034) and the patch from *Microsoft* came out.

The problem is similar to that seen with VBS/Bubbleboy and JS/Kak: *Microsoft* in its infinite wisdom once again classified a potentially dangerous ActiveX control as 'safe for scripting'. The control, which ships with *Office 2000*, is used by the 'Show Me' function in *Office* Help, and allows *Office* functions to be scripted. It was intended to automate demonstrations in help files by enabling the activation of common dialog boxes and selecting items on the dialog (including any of the checkboxes and the 'OK' button that usually activates the changes and dismisses the dialog).

The control's interface permits the scripting of any action in *Office 2000* that users could perform from the keyboard, including lowering the macro security settings. This action can be scripted from any HTML page viewed with active scripting enabled. The problem with this control was that abusing it could execute malicious active content regardless of macro virus protection settings.

Worm Components

The worm propagates from one location to another using several co-operating components. These are comprised of a JavaScript dropper and an *Office 2000* document used for email propagation that also serves as the dropper of a VBS file. The VBScript component, which contains the destructive payload, serves as a dropper for the joke HTML file and the HTML joke component displays a dialog box almost endlessly.

Worm Dropper

The original dropper is a simple HTML page which contains a JavaScript code that abuses the UA ActiveX control. Why is JavaScript used, when the other scripting component is written in VBScript? Most probably because the demonstration example accompanying the security warning about the UA vulnerability was written in JavaScript and the author didn't have what it takes to migrate it to VBScript.

The dropper activates the UA ActiveX components, and if it succeeds, drives it to display the macro security dialog box, and sets the security level to low before finally dismissing the dialog. In a separate frame the worm loads the second phase *Word* document component. A document called LD.DOC is downloaded from the Web server to the local PC and is then opened in *Word*. This duly executes the Document_Open macro.

Office 2000 Component

This component consists of a single *Word* document containing a macro. Initially, the macro drops the file LITTLEDAVINIA.VBS into the *Windows* system directory and registers it to run automatically at the next system startup. It does this by first creating the Registry key …\Windows\CurrentVersion\Run\littledavinia and pointing it to this VBS file.

After that, the component utilizes a well-known method for driving *Outlook* via its ActiveX programming interface to send the HTML email component (as the HTML body of the email messages) to all addresses in all address books. The worm avoids sending itself to the same address multiple times. If a user called Joe has already been targeted, the Davinia worm sets the value of the Registry key …\Microsoft\WAB\Joe to 1. Similarly, if the address book named Mybook is already processed, it sets the value of the Registry key …\Microsoft\WAB\Mybook to 1. Before sending itself by email to a given address, the worm checks for the presence of the appropriate Registry entry. If it is found, the email component will not be sent again.

VBScript Component

This component carries the worm's destructive payload. It is executed during the following startup after the infection. It drops LITTLEDAVINIA.HTML into the *Windows* system directory and sets the startup page of *Internet Explorer* to 'http://' which will result in a DNS error page during the next *IE* start.

Then the worm registers the above-mentioned HTML file to start up automatically during the next system start. That is, it would happen if the system remained in a bootable state, which it will not, as under usual circumstances the boot procedure would abort to the DOS prompt because of the tons of missing system files, most notably due to the lack of WIN.COM.

After all this, the worm starts up its destructive payload. It attempts to delete each file on local or mapped network drives, and then creates an accompanying HTML file containing the joke component with the name of the original file appended with an HTML extension. This way all files on the hard disk (except for the files in the root directory) will be replaced – the file is overwritten with the HTML, that file copied to filename.html (creating a ‘double extension’), and the original filename deleted.

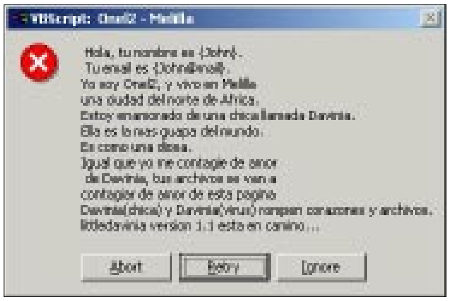
Due to obscure behaviour displayed by the *Windows* OS, some files will be processed several times, and will therefore receive multiple (3+) HTML extensions.

This clearly points out an enumeration issue in the *Windows Scripting Host* interpreter. When the script enumerates through the files in the current directory with a ‘for each’ loop (even a half-decent programmer would know that performing the enumeration this way is a very bad idea), the loop is executed until *WSH* finds it appropriate to stop. It is definitely executed more times than the number of the files originally in the directory. There are no strict rules, but the exact number of runs depends on the number of files in the directory with some slight deviation. The script obviously has great problems fighting MSCREATE.DIR – if it finds this file of zero length in a directory, it will create 40 files (each of zero length), with increasing numbers of HTML extensions. The last one has probably the maximum possible filename length.

Some overwritten files are re-processed in the later runs. Davinia definitely processes the files that were originally in the directory first, in the order that they appeared in the original file list. The condition upon which the enumeration stops is as yet unclear. Further processing depends on how the operating system fills in the file entries. The entries are not filled in order: the first couple of new entries appear at the end of the list, and then they start to appear alternately at the beginning of the list, too. The second run of processing does not follow strict orders: some files could be dropped from this round.

HTML Joke Component

When this file is opened in a VBScript-enabled Web browser, it will display the following dialog box. This dialog is displayed 412391491135182251919112722144491 times, which, given the tolerance of an average user, can be considered an infinite loop. Note that this dialog contains the SMTP display name {John} and SMTP email address {John@mail} as stored in the Registry. Davinia stores them in the HTML file when the VBScript component drops it. Since, by the time these dialogs appear, the operating system is in a pretty much idle state, this payload is very unlikely ever to be experienced.


VBScript: Onel2 - Melilla

Hola, tu nombre es {John}.
Tu email es {John@mail}.
Yo soy Onel2, y vivo en Melilla
una ciudad del norte de Africa.
Estoy enamorado de una chica llamada Davinia.
Ella es la mas guapa del mundo.
Es como una diosa.
Igual que yo me contagie de amor
de Davinia, tus archivos se van a
contagiar de amor de esta pagina
Davinia(chica) y Davinia(virus) rompen corazones y archivos.
littledavinia version 1.1 esta en camino...

Email Component

The e-mail messages sent out from infected computers contain the HTML loader of Davinia. This component is actually an HTML format email message with very simple content. When the message is displayed in the preview pane of an HTML-enabled email client (*MS Outlook* is a good candidate) and the mouse enters the area of the message, a series of additional browser windows are started pointing to six instances of two different Web pages located on the http://www.terra.es Spanish Web server. As the user switches between the windows, the desktop can soon be filled with *IE* windows.

The contents of these Web pages were removed on 15 January. Given information from several sources the content of the Web pages were the same as the one described in the worm dropper component.

The Moral of the Story

Davinia did not shake the ground with its complexity, but it did point out several problems that are becoming increasingly crucial. There is a definite conflict between the general attitude in the security field and the general attitude in the AV field. The ‘white hat’ security experts find it acceptable to publish code examples that demonstrate how to exploit security holes, while AV experts avoid publishing any source code examples of ‘concept’ viruses at any price.

Davinia (as well as Bubbleboy and Kak) heavily relies on the source examples published on security Web sites. The philosophy of free information flow in the security field was born back in the days when most of the people using the Internet were experts. They took it seriously, applying each and every security patch instantly.

Nowadays, average click-and-go users will not necessarily apply *Microsoft’s* latest security patch (they don’t even find out about them), not even if it has been available for over eight months. Security experts should take into account the changes in the user environment, and revise their attitude and their code of conduct accordingly.

Davinia was discovered and identified by a Spanish anti-virus company. Their subsequent press release and alert appeared on the morning of Friday 12 January, yet samples were not forwarded to REVS before the next Monday, which made it rather difficult for other REVS participants, like us, to respond to anxious user queries.

REVS was established to avoid situations in which vendors hold back samples in order, apparently, to achieve marketing advantages. REVS will not be taken seriously until we, its members, take it seriously. If we do not follow our own code of conduct, REVS will disintegrate from the inside.

OPINION 1

PHP go the Script Viruses

Denis Zenkin & Mike Pavlushchik
Kaspersky Lab, Russia

The beginning of 2001 brought the computer industry yet another surprise when an unnamed US anti-virus software reseller announced the discovery of PHP/NewWorld – allegedly the world's first computer virus made using the PHP script programming language. The surprise consists of two parts. Firstly, this was indeed the first announcement of PHP viruses widely reported by the world's mass media.

Secondly, the announcement contains incorrect information, because the very first PHP virus (PHP/Pirus) was actually discovered several months ago in October 2000. Nevertheless, the announcement made a great impact and many AV tech support departments received numerous calls from frightened people requesting a clarification of the issue.

What is PHP?

Let's begin at the beginning. PHP (Hypertext Preprocessor) is one of the most widely used server-based script languages which allows users to create and integrate script programs into Web sites. On 10 January 2001, the *Netcraft* company (www.netcraft.com) reported that there are more than 5 million Web sites using PHP all over the world. PHP enables Web developers to write dynamically generated and highly customizable HTML pages easily.

PHP scripts operate in a similar way to other script programs. This can be illustrated by the following simple example: depending on the current date, Web site visitors are shown a customizable message. PHP allows for full automation of that procedure making it unnecessary to change the message manually every day. To do this, a Web developer need only build a corresponding PHP script into the HTML page. When a user visits this page, the Web site automatically sends a request to the PHP processor installed on a service provider's server. The PHP processor checks the current date and generates an HTML page containing the corresponding message. Then the Web site shows this page to the visitor (see the diagram overleaf).

Much of PHP's syntax is borrowed from other programming languages – C, Java and Perl. It also has some unique features. This hybridization provides a script language ideally suited for operating on small and medium-sized Web sites. PHP has some advantages over other script languages, while keeping much of the functionality needed for site construction. It is much easier to understand than Perl, faster, more stable and less resource-intensive than Cold Fusion, and runs on nearly every platform unlike ASP or Visual Basic Script (VBS) created especially for Internet Information Server (IIS) and *Windows* environments.

PHP is distributed free of charge and is available for download on the PHP Group Web site (www.php.net) or any other PHP network member's Web site. In addition, one can easily obtain PHP source code from these sites, rendering PHP very flexible and useful for developing highly customizable solutions for user-specific tasks.

How do PHP viruses work?

Both of the currently known PHP viruses work in the same way. They gain access to the current directory (PHP/Pirus) or the C:\Windows directory (PHP/NewWorld), and infect files with .PHP and .HTM extensions (PHP/NewWorld also infects the .HTML and .HTT files). While infecting files, both viruses use a very primitive way of planting the virus code: they insert not the code, but only the 'Include' command to invoke the original virus file when the infected file is run:

```
<?php include(“[virus path & filename]”); ?>
```

Thus, the infected system has only one virus copy, while all other infected files simply refer to it. These viruses perform no other activities and have no additional payload.

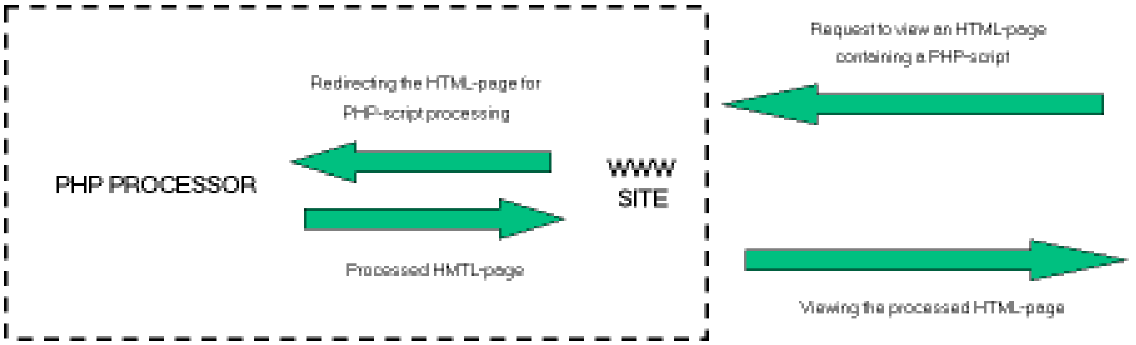
Are PHP viruses dangerous?

We conducted a comprehensive analysis of the known PHP viruses and researched the possible direction of their evolution. We concluded that PHP technology poses no current threat and cannot be considered dangerous even in the longterm forecast. This conclusion is based on the following findings.

Firstly, the likelihood of a PHP virus penetrating into Web sites and service providers' servers by its own means is negligible. This is due to the default settings of Web-based security systems which do not allow access to directories other than the parent Web site environment. Therefore, PHP viruses can successfully proliferate only within an infected Web site.

In other words, a PHP virus can penetrate a computer only if it is manually planted as a result of an administrator's intentional action or the external hacking of the system. The first scenario is unlikely to happen because the malicious person will be exposed immediately. In the second case, using a PHP virus is meaningless. Why would a hacker who has gained full access to an entire Web site only plant a questionable (in terms of its operation and effectiveness) PHP virus when he or she could do a lot more harm by simply destroying all of the information?

Secondly, even if a PHP virus succeeded in getting into a Web site by any of the aforementioned means, it would still have no chance of spreading further. It cannot gain access



to the system areas of the service provider's server and it does not have the ability to spread to other Web sites or PCs of the visitors who view an HTML page containing a malicious PHP script. This last case is not possible simply because a user receives a pure HTML page with absolutely no scripts inside from the PHP processor.

Having said that, we can anticipate a PHP virus which is able to instruct a PHP processor to insert a PHP script in the HTML-page transferred to a user which will later be executed by the locally based PHP processor. On the other hand, such viruses will have no chance of becoming widespread simply because a PHP processor installed on a home computer or a workstation within a corporate network is a singular phenomenon – PHP is now popular only as a server-based script language. The absolute inability of PHP viruses to spread under normal conditions confirms the impossibility that they could ever be 'in the wild', and renders meaningless any further research of virus writers in this area.

Are we expecting other PHP viruses?

Despite the arguments above, we do not recommend the belief that PHP scripts are totally secure and harmless: their armoury contains all the features necessary to perform the most dangerous tasks like changing and deleting files, sending out confidential information, email distribution, etc. In addition, PHP scripts do not require compiling and are available in source code. This enables a malefactor to modify the existing code and learn how to create new PHP viruses. The history of computer viruses is full of examples of an application or a platform which used to be considered absolutely safe suddenly becoming a nightmare for users.

The main danger that could stimulate the development of PHP viruses is the discovery of some serious security breaches in the PHP environment. This could be the starting point of a brand new breed of PHP viruses exploiting a specific breach. An example of how noxious this could be is the 'Scriptlet.Typelib' vulnerability in *Internet Explorer*. This breach allows viruses to spread via email without attached files and to penetrate computers right after an infected email is read. Although *Microsoft* released a patch in November 1999, the Kak worm that exploits this vulnerability is still in the top ten list of the most widely spread viruses. This shows that users often ignore software vendors' advice to keep their applications up-to-date. Under certain conditions, this could happen with PHP as well.

The availability of PHP source code has a double meaning: on the one hand, the source code significantly simplifies searching for security breaches – one does not need to decompile the code in order to study software's internal structure. On the other hand, thanks to the source code, users can effectively patch security breaches without delay when the developers release a correspondent update.

If PHP technology should become a worldwide desktop standard (like *Windows Scripting Host* in *Windows*), a PHP-specific Internet worm could pose a real threat. This programming language has all the features necessary to deal with email. Just like the LoveLetter virus written in VBS, a PHP virus could gain access to email and, unknown to a user, send out its copies. However, the significant spread of PHP worms is next to impossible. Unlike the VBS processors that are installed by default with every *Windows* installation, the presence of PHP processors on workstations is a rare occurrence. Hence, most users would simply not be able to start an infected PHP file.

The third possible way of misusing PHP technology could be multi-component viruses that have a PHP virus as one of the modules. The carrier module could be of any executable type, e.g. EXE, COM, VBS etc. When the virus is started it could check for the presence of the PHP processor and if it were found on a computer, the virus could drop an additional PHP virus in order to make virus detection and removal more difficult.

Conclusion

All the currently known PHP viruses (and there's one PHP Trojan – PHP/Sysbat) pose no threat, and it is impossible for them to get into the wild. This is not simply because they do not have sufficient capabilities, but also because all the major AV vendors have already updated their products to deal with them. The potential threat of future PHP viruses is considered to be low simply because PHP technology is unlikely to become a desktop standard. The strange fate of Java viruses serves as an excellent paradigm: since 1998 only a couple of Java viruses have been discovered; none of them has appeared in the wild, and now they are only present in private virus collections.

OPINION 2

Apple of Discord

Costin Raiu
Kaspersky Lab, Romania

Mac *Office 2001* is a relatively new product, to which, when released in October 2000, I paid little or no attention. This was mainly because I do not use or own a Mac computer and, well, because I was not expecting it to be anything special.

The first wave of news that this was indeed an out-of-the-ordinary release came about a month later, when Igor Muttik of *NAI* informed us that the new file format brings more trouble than expected. It was discovered that a sample of W97M/Proverb.A, otherwise a pretty common virus, was not detected any more by most AV products in a Mac *Office 2001* sample, while the virus was perfectly able to replicate without problems in *Word 97* or *2000* on *Windows* systems.

At the beginning, this was believed to be caused by some major change in the file format, but soon it became clear that the relevant change in the file format was only 2 bytes long, or more precisely, 2 bytes short. When comparing the module holding the virus in Mac *Office 2001*, and a similar module in Mac *Office 98* (the previous Mac *Office* release) it was noticed that one field which is a DWORD (32 bits) in Mac *Office 98* modules was now only 16 bits long in Mac *Office 2001*. Since most anti-virus products were simply unaware of this change, they failed to detect the virus in its new instance.

It was later found that the 2-byte change in Mac *Office 2001* was unknown even to *Microsoft*, and it was caused by a different compiler used for the new version.

The Changes

Both Mac *Office 98* and *2001* store macros in VBA5 format. I was pretty surprised to see that Mac *Office 2001* is also VBA5. I was expecting to find VBA6 implemented in the new 2001 release of course, but apparently someone decided that they should stay with version 5 of Visual Basic for Applications for the new version of the *Office* suite for Mac systems.

One of the reasons for my assumption was the following phrase, taken from a Mac *Office 2001* press release issued by *Microsoft* on 19 June, 2000: ‘Office 2001 sustains seamless compatibility by sharing the same file format as Office 2000 for Windows.’ As you can see, press releases are not always accurate.

Regarding the VBA5 from Mac *Office*, the internal version number that can be found in the _VBA_PROJECT stream of a VBA storage is 0x62. Other common values include:

Product:		**VBA Version used:**
Office 97	(VBA5)	0x005E
Office 2000 Beta (1?)	(VBA6)	0x006B
Office 2000 Release	(VBA6)	0x006D
Early Beta of O2K?		0x0065
MacOffice 98	(VBA5)	0x0060
MacOffice 2001	(VBA5)	0x0062
Office 10 Beta 1/2	(VBA6)	0x0070

It appears that looking at this number is the only way to determine if the modules from a VBA storage are in Mac *Office 2001* format, or more precisely, to distinguish them from those created by Mac *Office 98*.

I say that because the modules themselves are extremely similar to Mac *Office 98* modules. If you are not familiar with them, they are basically VBA5 (e.g. *Office 97*) modules, but with most of the information stored in big-endian format. For example, the well-known FE CA 01 00 marker for the p-code line table is CA FE 00 01 in the case of Mac modules.

However, the important difference between Mac *Office 98* and *2001* can be found in one of the many variables present in a VBA module, as you can see below:

		F1	F2	F3	F4
PCO97:	(ofs1:)	(A3 00)	(00 00)	(88 00 00 00)	(08)
MacO98:	(ofs2:)	(00 A3)	(00 00)	(00 00 00 88)	(08)
MacO2001:	(ofs3:)	(00 A3)	(__ __)	(00 00 00 88)	(08)

The three hex strings above are extracted from a VBA module which was subsequently saved on PC, Mac *Office 98* and Mac *Office 2001*. Obviously, the strings are extracted from different offsets for each module, but listing the exact offset in here would not make any sense without listing the entire modules, so I just put ‘ofs1’, ‘ofs2’ and ‘ofs3’ in front of the three strings.

As you can see, the ‘F1’ field is basically the same for all three platforms, in this case ‘0xA3’, but for Mac versions it is stored in big-endian format. So far, nothing special. Now comes the tricky part: ‘F2’ is present in *Office 97* and Mac *Office 98*, but it is missing from Mac *Office 2001*! The other two fields, ‘F3’ and ‘F4’ are practically the same.

Therefore, if you parse the VBA modules structure by structure in order to reach the p-code line table or the compressed source you should take notice of this change.

Interestingly, I was informed that some products simply do not care about all the structures in the VBA module, and simply scan for ‘FE CA 01 00’ or the equivalent big-endian version in order to find the beginning of the line table. Such products do not need any special check for Mac *Office 2001* compared to Mac *Office 98*, but the method is probably not very reliable either.

The compressed source of Mac *Office 2001* modules is no different from that of the Mac *Office 98* format. The only interesting thing is that the traditional PC ‘0D 0A’ LFCR sequence is ‘0D’ for Mac. Since the ‘dir’ stream format is exactly the same as the one in PC *Office's*, products which use the compressed source for detection of macro viruses should not have any special problems with detecting instances of viruses saved in Mac *Office* formats – unless they happened to use ‘0D 0A’ as an end-of-line marker, of course, as I did.

The executable code form of VBA macros, the so-called ‘execodes’, also present a very interesting change in Mac *Office 2001* (and *98*). As some of you may know, the *Office* developers implemented a very simple ‘compression’ scheme for VBA5 or VBA6, thus reducing the size of disk space required to store the __SRP streams. They took the first 246 most widely-used opcode values which were natively 2 bytes long, and translated them into the 1 byte values 0..0xf5.

Then they used some of the remaining 1 byte entries [0XFB..0XFF] to create pairs of the form (0xff [0..0xff]), (0xfe [0..0xff]) and so on, thus achieving a simple means of compression for the values with the highest probability of appearing.

However, the interesting part comes in Mac *Office* execodes. The compression/decompression wrapper written by the PC *Office* developers seems to be missing from the Mac *Office* release so the execodes are not compressed at all, but found in their native form! For example, the extremely common PC execode 00h (‘EX_Bos’), which is 1 byte long and which is internally translated by the wrapper into the native execode value 0x1C4, is found in Mac __SRP modules as ‘01 C4’ which is exactly the big-endian instance of 0x1C4. The same goes for the other translated execodes.

Therefore, those who scan the __SRP modules for strings (even containing wildcards) in order to detect viruses like X97M/Jini.A will have some problems detecting their Mac *Office* forms, if they ever happen to appear. I say ‘if they ever happen to’ because virus samples containing only execodes such as the infamous X97M/Jini.A sample are most likely produced by anti-virus engines and do not simply appear naturally. (If your anti-virus engine fills the line table with empty entries and overwrites the compressed source with an empty module named ‘Module1’, please contact me; craiu@pcnet.ro) Moreover, a sample containing only PC execodes, with no source or p-code, will not work on Mac *Office* and vice versa – the price of ‘compatibility’, I dare say.

However, those *anti-virus* products which include true execode parsers, and at the time of writing I am only aware of one, will only have to disable the translation routines in the case of Mac *Office* macros and, of course, update the __SRP module parsers to deal with the big endian values. Another change that differentiates Mac *Office* from PC Office can be found in *PowerPoint* presentations. PC *PowerPoint* presentations keep the VBA storage in the ‘PowerPoint Document’ stream in an atom marked with the WORD ID value 0x1011. At binary level, after the ID WORD comes a DWORD containing the size of the atom data. As I said, so far so good. Next, on PC *PowerPoint* presentations we should find the size of the uncompressed VBA storage, because the VBA storage (which is an OLE2 file) is stored in compressed form, using the well-known ZIP ‘deflate’ algorithm.

However, again for some unknown reason, the Mac developers did not (want to?) have access to the same code as the PC *Office* developers, so in Mac *Office PowerPoint* presentations the VBA storage is kept in uncompressed form. Moreover, the DWORD storing the uncompressed size of the VBA storage is missing. The following image shows the difference:

	F1	F2	F3	F4
MacOffice:	(11 10)	(00 1C 00 00)	(D0 CF 11 E0)	(A1 B1 1A E1)
PC Office:	(11 10)	(25 2C 00 00)	(00 70 00 00)	(78 9C EC 5A)

(A beer at VB2001 for the first person to tell me which virus *sample* the PC *Office* string is extracted from!)

F1 is the same – the 0x1011 marker – and F2 stores the size of the atom. F3 holds the size of the uncompressed VBA storage in PC *Office* but is missing in the Mac *Office* presentation, where in F3 the first 4 bytes carry the well-known OLE2 signature ‘DOCFILE’. So there is yet another place where the engine code needs to be (if not already) updated. However, it is not that hard to check if F3 is 0xE011CFD0 or to check if the *PowerPoint* presentation was saved on Mac *Office* (the change is no different for Mac *Office 98* and *2001*).

Learning Lessons

Extra care should also be taken when cleaning Mac *Office 2001* (and *98*) viruses. Usually, it will not suffice to update an engine to support detection of the Mac *Office* formats. Disinfection routines will probably have to be updated as well, of course, depending on how ‘deep’ your disinfection code goes.

Most of the important structures, such as the 1/0Table streams for *Word* documents or the Workbook stream for *Excel* documents, and even the ‘dir’ stream of the VBA storages, have basically the same format for PC and Mac *Offices*. However, those products performing per-module disinfection through various methods which touch the modules themselves will probably have to work on the disinfection code a little bit.

The lesson we all have to learn from the Mac *Office 2001* instance of W97M/Melissa.W is very important. First, new file formats should not be overlooked. As I was saying, most of the industry was aware of the problem in November 2000. Secondly, we should keep an eye out for those non-PC program releases. Say, if we ever hear of an *Alpha* or *SunOS Microsoft Office* release …

FEATURE 1

Good Korea Move

Hyunwoo Lee
CERTCC-KR, Korea

CERTCC-KR is an institution that performs the role of a Computer Security Incident Response Team (CSIRT) in Korea, coping with the spread of computer incidents and taking preventive measures against them. Recently, it has also taken on the role of a Computer Virus Incident Response Team (CVIRT) to deal with the recurring damage caused by computer viruses.

This article reveals that incresingly, attacks are made by malicious agents using Internet worms and Trojans. Recent attacks using DDoS agents, Internet worms and Trojans exposed many problems with existing security systems and response mechanisms. We need more comprehensive and preventive measures to enable us to defend ourselves against potential future threats.

Traditional Attack Methods

General attack procedures for system intrusion can be classified into three steps. The first is an information gathering phase – collecting information on the network topology, OS fingerprint, network devices and network services of a target. It is usually performed using automated scanning tools such as nmap, hping, sscan and mscan.

The second step is a penetration phase – actually getting into the system using the information collected in the information gathering phase. Widely known bugs such as buffer overflow and insecure configurations on the network server are used to penetrate the target machine.

The last step is an attack extension phase. In this phase, a number of Trojan horses and backdoors are set up in order to eliminate the trace of the penetration, make re-penetration easy, and extend the penetration across other systems.

New Attack Trends

The biggest motivation for changing traditional attack methods comes from the enhancement of the defender's security levels. The widespread use of firewalls and intrusion detection systems (IDS) provides very effective countermeasures against traditional attacks. Also, the cooperation among CSIRTs worldwide is narrowing down the activities of intruders. However, many techniques and tools for advanced system intrusion are actively developed, publicized and widely used to overcome these barriers.

A distributed attack, attacking one or multiple target networks from many systems, gives an intruder more information on target networks in a short time span. It also uses forgery attack patterns to hide the real attack patterns so that the information acquired by IDS is rendered useless. Furthermore, using tools in the form of distributed agents such as various DDoS agents, an attacker can easily take control of many agents remotely to attack another system without getting into a system again. This reduces the chances of detection.

As homogeneous network structures are deployed and only a few systems and applications are used on most Internet sites, just one of vulnerability on this system or application can play havoc with many organizations. For example, various RPC-related vulnerabilities found in the *Solaris* system were widely and effectively used to hack into many Internet sites worldwide.

This uniformity of the Internet environment eventually led to the introduction of automated attack tools such as Internet worms. The ADM Internet worm (ADMw0rm), the Millennium Internet worm and the Ramen worm have been found in the wild in 1998, 1999 and 2001 respectively. Numerous automatic or semi-automatic attack scripts have been found in recent incidents, and they are sometimes used as a means of spreading DDoS agents such as trin00 and TFN. These tools make it possible for an attacker to execute a parallel attack and extend the scale of an attack.

Another tactic is a backdoor, a technique which allows intruders to get back into the system whilst avoiding any authentication process and thus leaving no trace. But the traditional backdoor technique has been widely recognised and can easily be detected by most security systems. So the backdoor technique has been improved.

Recent backdoors found in the wild do not use specific ports or connections to get back into system. Instead, they use a remotely controllable agent using a covert channel so as not to be detected.

Since this covert channel technique can be implemented on various protocol layers such as ICMP, UDP, TCP, as well as various application layers such as http, DNS and email, which are usually open to the public by security systems, it also provides methods for bypassing firewalls and IDS. In addition, it has encoding functions to bypass IDS. It is just like an encrypted virus being able to bypass a virus scanner.

The techniques used to hide an attacker's activities on computer systems have also been improved. Rather than simply using a Trojanised version of login, ps and ls programs, covert functions are implemented into the kernel level. The kernel backdoors for various Operating Systems and run-time kernel patching techniques have been publicized widely. The forms and the functions of the backdoor are becoming diversified. Normal backdoors take the form of servers so that an intruder can connect to the backdoor or

Trojan horse. But they are moving into client form such as reverse telnet and reverse ssh to bypass existing security habits which do not filter outgoing packets. Moreover, all these techniques are increasingly applied to Internet worms and Trojan horses.

The number of *Windows*-based attack tools is increasing. This is because the security system can be bypassed by attacking the average user who has no security knowledge. Also, the increased computing power of the PC makes it attractive to attackers. Nowadays, *Windows* client systems can be attacked very effectively and used by means of attack tools in the form of agent. An agent can operate on behalf of an intruder to help attack another system and give the intruder feedback in various ways without getting into the system. Tri00, originally developed in the *Unix* environment, is ported to *Windows* systems in this context.

Recently, such agents have tended to combine key features from viruses, Internet worms, Trojan horses and even agent functions used in the field of Information Technology. The propagation, information gathering, remote control, distributed attack and automatic update functions are all getting integrated into attack tools. The MTX virus illustrates this phenomenon very well.

Social engineering is an essential element in the case of attacking a well-secured site or in preparing for a large scale attack. Since social engineering attacks tend to be invisible and overlooked by many security measures, it poses serious threats. The Melissa virus is a typical example of an attack using social engineering. It uses the email attachment as a transportation mechanism to bypass a security system, taking advantage of the fact that many organizations allow incoming email attachments. It also exploits the trusting relationship among email users by using the email address book as a means of spreading.

Another interesting case is that which utilises Trojan horses such as BackOrifice or Netbus. They provide script kiddies and wannabe hackers with interesting functions to play with, and consequently they are dispersed worldwide. Now, an attacker has enough computer resources worldwide to perform a massive attack. Social engineering attacks are unpredictable in their forms. So, it is very difficult to detect them. We have to keep in mind that a network attack need not only be made by technical means.

Countermeasures

The new attack methods described so far have evolved as a result of intruders trying to find the weakest point in the network. This transition arises as the common issue in the anti-virus and anti-intrusion areas. This is the reason why anti-virus software and IDS have some functions in common, though their detection techniques are different.

So, could we find more effective response measures against recent virus and worm incidents than those taken in the area of anti-intrusion?

Since experiencing a DDoS incident in early 2000, there has been a significant change in the anti-intrusion community in responding to incidents. Great efforts have been made to prevent and detect incidents early by sharing information about the incidents – the Ramen worm was found and analysed through a public mailing list. The anti-intrusion community is focusing on network monitoring to detect unknown, new malware rather than relying solely on IDS. Many security experts have released technical guides to defend against specific problems in a timely manner. These kinds of response activities have helped users to have a sense of security, better response techniques and abilities to deal with new attacks quickly. The honeynet project (http://project.honeynet.org/) is a good example of this.

In Korea, *CERTCC-KR* released the *Scan Detection* program which offers detection of scan attacks as well as network monitoring functions. Currently, many domestic sites have installed it so as to report detection information directly back to *CERTCC-KR*. And this information reveals not only the status of attacks on domestic networks but also the spread of new Internet worms. An abnormal increase in scanning on port 25 led us to recognise that an email worm was spreading. We found the Detlog worm was spreading across domestic networks by investigating the system that scanned the Netbus port extensively.

However, the anti-virus community, mainly comprised of AV software vendors, has shown very limited activities in dealing with recent virus/worm incidents. While the anti-intrusion community makes an effort to find preventive measures in all aspects of security problems, the AV community tends to focus on the technically oriented aspect of a specific problem. In addition, some anti-virus software vendors give users a false sense of security, which misleads them into relying on anti-virus software as the sole countermeasure against these attacks.

This false sense of security has led us to experience continuous damages from many variants of Melissa, even though there have been mail server security guidelines and security tools to deal with such attacks in the anti-intrusion community. Recently, the response from anti-virus vendors to the Ramen worm has worsened this situation – users can easily misinterpret the Ramen incident and think that they are safe if they only update the virus signature once.

For more effective virus incident response, we need a community widely open to the public. This will promote awareness, techniques, detection of unknown attacks and other issues among the participants. To detect a new attack early, we should monitor network activities. As for preventive measures, technical and managerial guidelines should be developed and observed by all staff within each organization. The 'Trusecure Anti-virus Policy Guide' is a good choice as a template for anti-virus policy. When a new security incident occurs in the wild, managers want to hear that they are safe from this attack. Let's work towards implementing better cooperation and safer computing environments worldwide.

FEATURE 2

Hi Fidelity

Richard Holder
Fidelity Investments, USA

The idea of protecting the world's largest mutual fund company from computer viruses and other malware sounds challenging. The reality is even more so. Over 32,000 full-time, contract and temporary employees, 50 entrepreneurial business units, operations around the globe and a complex and fast-changing computing environment add unexpected dimensions to the challenge.

The Set Up

Fidelity Investments' growth has been fuelled by technology. In fact, technology workers outnumber fund managers, analysts and traders 18 to one.

The computing environment at *Fidelity* is broad and complex, consisting of about 40,000 workstations with hundreds of servers running predominantly *Microsoft* operating systems and applications. There is a significant mainframe and Unix presence plus a vast collection of internally developed, custom software programs running on various computing platforms.

It is all channelled through one corporate network with a single email system. In addition, Fidelity.com is used by almost 40% of the company's 16 million customers. It consists of dozens of tightly-woven sub-sites comprising 30,000 pages.

Fidelity's corporate model is decentralized. Each business unit operates like a separate business. This business autonomy can complicate enterprise-wide systems management, especially virus defence.

Add to the mix that *Fidelity* employees work in several languages, many cultures, numerous time zones and, most of all, differing information technology requirements from business unit to business unit and you have what could be the roadmap to a nightmare. But it's not. Challenging, yes, but not a nightmare.

The Players

Based in Boston and part of *Fidelity Corporate Security's Information Security Technology Group*, Corporate Virus Defence (CVD) ensures that this complex organization remains virus free. Their technological strategy uses multiple anti-virus products at strategic locations: firewalls, email gateways, servers and desktops. CVD is also the single source for all virus information at *Fidelity*, implementing anti-virus policies and advising business units on procedures to keep problems at bay.

Richard Holder is a key member of *Fidelity's* Corporate Virus Defence Team

The Virus Response Team (VRT), which includes members of CVD, manages the response to actual virus threats. When a new virus or worm is discovered, the VRT gathers an assessment team (including representatives from information security, desktop, server, messaging, firewall, telecommunications, help desk, international and communications) to determine potential risk to the firm.

Once the VRT validates the threat, they analyse the virus' characteristics including spread rate, delivery mediums, damage and possible security breaches (such as password stealing) and the potential risk to *Fidelity*. The VRT also reviews the countermeasures in place and customizes a short-term plan to ward off the virus.

The VRT manages the plan with assistance from a distributed organization of information technology professionals throughout *Fidelity*, as well as designated Information Security Officers (ISOs) from each business unit. Updates are sent through numerous communications channels including worldwide conference calls, alphanumeric pagers, email, internal network news updates and pre-recorded telephone messages.

The story often ends with updates to anti-virus measures followed by an 'all clear' and a sigh of relief from everyone involved. But, in the unlikely event that a virus gets inside, an emergency virus response plan is activated.

The VRT works with business unit technologists to segregate outbreaks. In an isolated lab, anti-virus definitions and repair utilities are tested against live copies of the virus. The tested tools are then shared with other business units and certified before being rolled out across the enterprise.

At the same time the VRT coordinates a multi-channel communications campaign for users with instructions on how to stop infections at the desktop. Status updates are gathered on conference calls and all plans are reviewed and modified as the incident unfolds.

In the past, these events have lasted anywhere from a few hours to a couple of days. Once the immediate emergency passes, the VRT is free to conduct a root-cause analysis, where it compiles lessons learned and looks for ways to improve future emergency processes during an after-action analysis.

The Plan

Fidelity's Corporate Virus Defence program depends on teamwork with all *Fidelity* business units. All anti-virus software is tested and certified with other business units and included in consistent workstation and server builds distributed throughout the enterprise.

The business units also have a hand in updating policies and standards, which are published on the company Intranet. Business Unit Information Security Officers and Contingency Planners are instrumental in ensuring that anti-virus policies and standards – including anti-virus software configuration, definition update frequency, scheduled virus scans, centralized reporting and problem escalation – are consistent throughout the organization.

Virus Response Team members and incident responders are encouraged each year to take a *Fidelity*-sponsored anti-virus training program with topics including technical response, escalation and virus awareness. But user training actually begins on each employee's first day, as virus awareness is an important part of the *Fidelity* new-hire orientation program.

An emerging area of concern for Corporate Virus Defence and for teams of this nature throughout the business sector is how remote access to email and the network can co-exist with the virus threat. The number of notebooks for use on the road is exploding. Plus, employees can get basic access to their email account from a home PC.

Fidelity's isolated lab is ideal for virus testing and analysis

Downloadable virus software, virus definitions, patches and virus repair tools are available on an Intranet site for the use of all the company's employees. Anti-virus software licensed to *Fidelity* for use at work or home is also available to employees upon request and free of charge. This provides an additional layer of protection for the use of home and remote PCs.

The Prognosis

We are not complacent in our struggle for protection. The future of virus fighting at *Fidelity* includes:

- Strengthening virus defence over infection response
- Reviewing and testing new anti-virus technologies as they are released
- Continued analyses of future virus threats, especially those with damaging payloads, widespread infection capabilities and targeted attacks against specific businesses or industries
- Particular attention to mobile code (e.g. Java and ActiveX) and their increasing threats
- A continuously watchful eye on threats to PDAs and cell phones
- A better understanding of how to deal with distributed networks of malicious code (W95/Sonic and VBS/LoveLetter).

When all is said and done, the greatest allies any corporate virus fighter can have are senior executives. Fortunately, *Fidelity* management understands the business need for an effective virus defence program.

With solid support from our management and business unit partners we have been able to implement anti-virus products into our standard builds, scan Internet email messages and block certain types of attachment. These are the solid foundations on which we have built our strong event management plans and, taken together, they have yielded significant results.

The Plea

While *Fidelity Investments* has gone, and continues to go, to great lengths to protect itself in this field, we would still like to see some improvements within the anti-virus community itself. It has often been said but we repeat that a single virus-naming convention with industry-wide cooperation would be a good first step. We are also actively advocating more effective legal remedies against virus writers, both domestic and foreign.

Having said that, it is incumbent on each company and corporate organisation to set up their own defences against the growing virus threat. We are confident that with management support, cross-company teamwork and a little creativity, every company can be more effective in holding the bugs at bay.

PRODUCT REVIEW 1

Aladdin eSafe Enterprise v3.0

Matt Ham

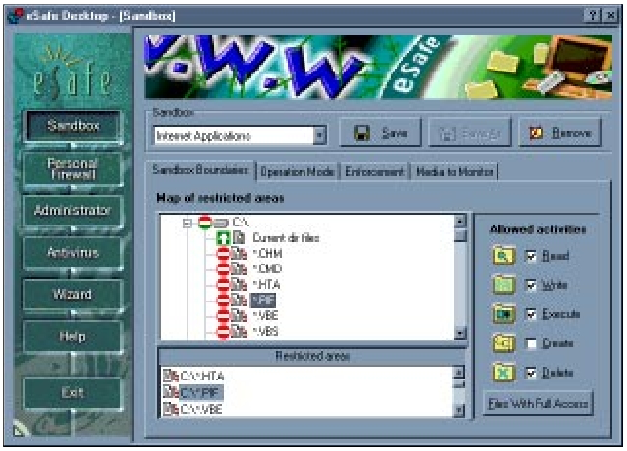
Aladdin Knowledge Systems' eSafe Enterprise was among the first products I ever reviewed for *Virus Bulletin*, and at that time it had a more or less unique combination of features. The integration of general security and anti-virus protection is a field which has since become home to a wide variety of players. Despite this, *eSafe Enterprise Enterprise* (and *Desktop*), the subjects of this review, are still in many ways the most obviously designed along the lines of inseparable security and AV.

The product's feature set has been refined and tweaked over the intervening years and the overall detection rates have improved greatly – making this a good time to revisit and inspect the resulting incarnation. The treatment of some aspects of the protection offered will be brief (due to space restrictions) and to the point, which should not be taken to mean that they are unimportant, more that viruses are the prime concern as far as this review is concerned.

Contents and Documentation

Most of my testing was performed on an electronically updated copy of *eSafe Enterprise Desktop* though a full boxed copy of *eSafe Enterprise* was available for the study of packaging, network features and documentation. The box is a sturdy creation in royal blue – a great improvement on the old packaging – and fully packed with documents of various sizes.

Again, the documentation has been subject to some serious overhauling and is larger both in range and bulk. The two most obvious items, the CD and main manual, are accompanied by a White Paper on 'Safe Internet Connectivity for the Enterprise', a Quick Start Guide, the *eSafe* product range leaflet, a registration card and pocket guides for both Admin and Users.



The two guides differ in that the User Guide is covered in a wipe-clean surface – presumably in case the Administrator is cursed with particularly drooling users. The information within both booklets is good and well-directed, with distinctly less assumption of insider knowledge in the User version as opposed to that of the Administrator.

The Quick Start Guide logically progresses in complexity from the reference card for Administrators and, as expected, contains much more detail about the installation of the software, including all manner of things to watch out for and factors to consider when deploying and configuring.

Registration is necessary for updates, upgrades and technical support – and although a card is supplied, the process can be more speedily performed electronically. The White Paper and product leaflet are both ultimately advertising material, though the former is actually quite interesting to read, despite its agenda of making the Internet seem as nasty a place as possible.

This leaves the product manual, which stretches to three hundred pages. This is, and at that size should be, a source of detailed information about all that is *eSafe Enterprise*. The style is very similar to some of the self-tuition books popular at the moment, even to the presence of having review multiple-choice questions at the end of each chapter.

The manual does not seem to be lacking in any major field, an index and appendices adding to its usefulness. Overall, it seems well written in a style less dry than others I have had the pleasure of reading. Most notable points of amusement for me were a typo referring to 'denial-fisheries attacks' and the intentionally comic wrong answers to some of the review questions, including 'vandals' in pin-striped pyjamas. With the non-software portion of the package thus appreciated, the testing of software was the next step – would this progress as smoothly?

Installation and Upgrade

The installation options for *eSafe Enterprise* are varied with central administration available through the 'eConsole' application. In this case, a standalone installation – the *eSafe Desktop* portion of the product – was used for the test procedures, but installation across *NT* and *NetWare* network deployments is fully supported and other networks can be used with more manual tweaking.

The choice of languages, in this case pretty extensive, and the licence agreement form the traditional start of the installation process, followed by the similarly ubiquitous choice of custom or standard installation.

Custom installation allows selective installation of the Sandbox, FireWall, and on-access scanner modules and offers the option of a scan preceding the completion of installation. After this, a reboot finishes the process.

Two methods of update were available – the more useful for lab work being the use of a manually downloaded EXE file in order to update the virus databases. Updating within the program seemed to take longer to connect to the site, possibly due to version checking being performed, but other than this delay it had much the same effect as executing the standalone update.

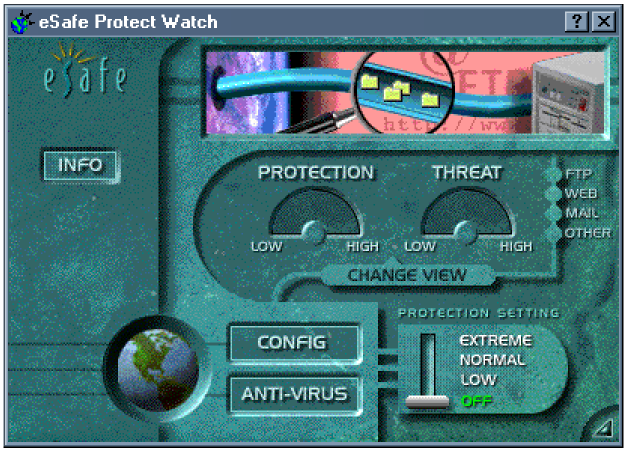
General Features

As mentioned previously, the range of activities performed by *eSafe Enterprise* is far from limited to the scanning for viruses that is the subject of tests at *Virus Bulletin*. These are all centrally configurable and enforceable over a network, or they can be configured from the desktop on standalone installations.

Since many of the operations involved are more related to security than to anti-virus, the central enforcement aspect is vital rather than simply optional. Present but untested is a Personal Firewall component, which allows for the standard IP address and port-blocking functions associated with the average firewall. There are, in addition, content filtering and blocking functions for URLs, newsgroups and data within the Personal Firewall. This includes a component which blocks certain contents but can also enforce the encryption of specified sensitive subjects, though this is a feature which would seem only to differ from content filtering in the nature of warnings given.

The more important (from an anti-virus viewpoint) extra feature of *eSafe Enterprise* is the so-called Sandbox feature, combined with the Privileges functions of the Client program. This combination is a policy enforcer, which offers control over a wide range of operations. Of these most of the Privileges are security related, such as disabling booting to DOS, operation of the Control Panel, Registry editing or the use of network drives. Clearly some of this control is less necessary on a well-administered *NetWare* or *NT* network than where clients are running, for example, *Windows 95*. Some, such as the Registry control, would be of general application in the prevention of many viral side-effects. The Sandbox, on the other hand, assigns similar rights to applications and is more directed towards virus control than to amelioration.

The Sandbox allows the outright banning of, or alerting the user to, operations considered to be either the work of viruses or more general malware, here termed 'vandals'. Operations can be banned in a blanket fashion – not so useful when operations include file operations of all kinds – or on an application or directory-based rule-set. Many popular applications are already configured with their normal range of operations and where they occur, while new applications can be set with custom parameters. The major problem with the setting process is that in order for it to be truly effective the process relies on a good knowledge of what an application might be expected to do legally. This is thus a portion of the product which assumes at least a little administrator skill for total effectiveness.



Scanning

When it comes to the scanning features of *eSafe Enterprise Desktop* my reviewer-style niggles start to appear. These are simply that the methods used to alter settings and perform scans are somewhat more complicated than seems absolutely necessary. From the main start screen as seen above, configuration can be performed from either the anti-virus or configuration button, so far, so simple.

Pressing 'Config' leads to the Configuration Wizard, which is mainly concerned with the deletion of various caches on bootup and the applications which may need to be Sandboxed. It is, however, possible to divert to Advanced Configuration at the point of entry to the Wizard, leading to the configuration screen on the opposite page.

From here the On-Access, On-Demand and Environment tabbed menus for the scanner can be reached by selecting 'Antivirus'. This is quite long-winded, and the On-Demand scanner can also be reached through the 'Anti-virus' button on the main start screen. The On-Access, On Demand and Environment settings are also reached through a start menu icon, though in this case via a different intervening interface. So, there are three methods of reaching the On-Demand scanner, and two each for On-Access and Environment settings, each using different names and GUIs. This was confusing but, thankfully, the settings themselves within the scanners and environment dialogs were straightforward and self-explanatory by comparison.

The Environment settings menu controls the anti-virus functions and provides information through three tabs, Paths and Messages, Virus Information List, and Password. In Paths and Messages the alert file, quarantine area and SmartScan file name are defined. SmartScan is an integrity checking portion of both on-demand and on-access scanners allowing faster scans of unchanged files which have been registered as clean in these SmartScan files. Notifications are also covered in this area, with sound alerts an option in addition to the more usual customisable text message, while a sub dialog allows for exclusion of specified files. The virus information list is a standard virus library, lacking only in detailed text descriptions of the viruses within it. The Password feature allows locking of access to configuration settings.

More direct control of the on-demand scans comes from the On-Demand settings menu. This allows for alterations to configuration through Scan Map, Scan Properties, Report File, Response and Schedule tabs. Scan maps is the area for the selection of targets to be scanned and the Schedule area allows for scheduling by the day week or month. Response can be altered for a removable virus, a non-removable virus or files which have failed the integrity check.

Quarantine cannot be set here but is configured in the environment area. Deletion, notification or user choice are offered as appropriate – with the option to disinfect or recalculate checksum files. The Report area holds no great surprises in store, allowing name and destination of reports to be changed with some size control features.

Finally, we reach the subject of the Scan Properties – which is a remarkably simple area. Apart from the control of checksumming only one option is available for scanning; whether or not to use the ‘analyze’ feature. This appears from the description given to be a behaviour blocking or heuristics system.

eSafe Enterprise’s On-Demand controls are slightly different from the On-Access ones – the third of the three settings dialogs. One tab here gives options of whether user interaction is required, whether SmartScan should be activated and which extensions should be scanned – the default varies from a list on-access while on-demand is all files with archive handling.

Scanning Activities are handled by the other tab – a fine-tuneable system of allocating a reaction depending upon the media scanned and whether the scanned object is tagged as viral, disinfectable or not, detected by SmartScan or displaying activities which are virus-like (this last being equivalent to the ‘analyze’ option for on-demand scans). Here the function is broken down into components, allowing specific actions on illegal renaming or interrupt tracing in a program to name two of the options.

Detection

So, we come to the matter of detection, the mainstay of *Virus Bulletin* Comparatives and still of note in these standalone tests. Since the recent *ME* test was on a very similar version of *eSafe Enterprise Desktop* to this one, it was expected that there would be few problems with the test-sets on-demand – simply an addition to the detection by dint of the intervening updates. No differences were observed when scanning using the ‘analyze’ option, which left one set of scan results each for the on-access and on-demand scanners.

On the on-access front, the tests were skewed in the last Comparative by the method used to intercept File Opens by the *eSafe* product. This uses special triggers for the scanning of OLE files, which were not set off by any of the standard *VB* testing methods. With a custom tool to circumvent such problems, the results on-access were expected to be better and indeed they were. No misses were seen in the ItW set and of the other misses (168 out of a test-set of 21,170 as opposed to 522 in the Comparative’s on-demand tests) were mostly polymorphics of some sort, macro or file. These figures were, overall, a great improvement as testified by the figures, with ACG.B, among others, detected here where it was not before.

The on-demand figures, however, illustrated a confusing feature. Misses were those seen on-access, plus some additions. Even more surprisingly, some of these new misses were detected in the Comparative tests, and have been for a considerable time. This had been noted by the developers independently, however, and a patched version from their Web site reduced misses to a very respectable 45 across all the test-sets. Those contained a sprinkling in the Standard set but were by and large polymorphic macro and file viruses once again.

Conclusions

The *eSafe Enterprise* range contains a virus scanner within a more comprehensive anti-malware and security package. Integrity checking, behaviour blocking and access control are all features which should minimise the impact of undetected viruses, while the scanner is certainly good and improving, if not yet perfect, on all fronts. The main hindrance as far as testing was concerned was the tortuous configuration method, not so much the configuration itself but the finding of it. This would probably be less of a problem to a standard user or administrator than to a tester, and the ample documentation mitigates even this to some extent. If you don’t mind complexity, however, the extra features might certainly be of interest to many administrators who can shield the complexities from their innocent users.

Technical Details

Product: *Aladdin eSafe Enterprise v3.0*.

Developer: *Aladdin Knowledge Systems UK Ltd*, Fairacres House, 2–3 Fairacres Industrial Estate, Dedworth Rd, Windsor, Berkshire, SL4 4LE, UK; Tel: +44 (0) 1753 622266, fax +44 (0) 1753 622262, email esafe.uk@eAladdin.com, WWW; http://www.eAladdin.com/.

Price: 25 users – 1500 Euros, 50 Users – 2500 Euros.

Test Environment: Three 750 MHz AMD Duron workstations with 64 MB RAM, 8 and 4 GB dual hard disks, CD-ROM, LS120 and 3.5-inch floppy, all running *Windows ME*. The workstations were rebuilt from image back-ups and the test-sets restored from CD after each test.

Virus Test-sets: Complete listings of the test-sets used are at http://www.virusbtn.com/Comparatives/NT/2000/11test_sets.html. A complete description of the results calculation protocol is at http://www.virusbtn.com/Comparatives/Win95/199801/protocol.html.

PRODUCT REVIEW 2

DialogueScience DrWeb v4.22

Matt Ham

DialogueScience's DrWeb has been a long-term contender in the *VB* Comparative Reviews, and of late has turned in consistently high detection rates and been given VB 100% awards with an increasing frequency. Having gained a VB 100% award in the last *ME* review, there were two choices open to me in this test; bulk the test-set out with new viruses or try some more experimental tests on the product. The viruses would in any case be present in the test-sets for the *Windows 2000* Comparative next month and so, with the developers keen to be put to the test, a pair of different tests were contrived. To discover quite what these were, of course, you will have to read on.

The Package

This review is slightly different from most other standalones in that a full boxed product was not received. *DrWeb* is distributed mainly in electronic format outside its native Russia, which made it more realistic and, given my knowledge of Russian, more feasible to do it this way. The first stop in this review is thus the Web site of *DialogueScience*, where such installations by necessity start. The Web site suffers from one problem – it is slow when connecting from the United Kingdom. Once connected, downloads progressed at a reasonable if not mighty speed, but to connect often took several lengthy attempts.

There is a choice of languages, defaulting to Russian, though German, French, Spanish and English are also available. The contents gain great praise for one feature – the mention of our Comparative Reviews in the news which makes up the main part of this page.

There is, however, another praiseworthy feature for users as well as *VB* reviewers, in that downloads are all easily accessed through the home page. A bar along the right hand side provides downloads of the entire product range, without the need for irritating registration or forms for such things as database updates. Since frequent updates are also available for definition files, this is likely to encourage the downloading of these files as it is a painless process.

This lack of unnecessary irritations is also seen in installation, which is a short and painless process. There are two main choices of installation, Compact or Standard – the latter adding a scheduler and a DOS version of *DrWeb* to the core components of *DrWeb*, *SpIDer Guard* and language support. The *DrWeb* component is the on-demand scanner for the product which bears its name; a custom feature allows independent selection of each of these components. After this short process and a reboot, installation is complete. The lack of configuration decisions at this stage does result in ease of installation, but this also means that all configuration options must be implemented after installation.

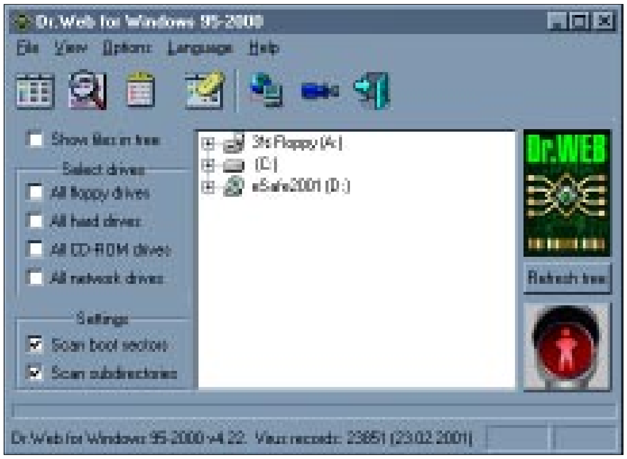
The Components

As mentioned, the *Windows ME* version of *DrWeb* consists of three components, the Scheduler being the only part of the package that does not come with *Windows* help resources associated with it. *DrWeb*, the on-demand component, is referred to at some points in the documentation as *Doctor Web* and the two designations are easily interchangeable.

SpIDer Guard is the on-access component and presumably owes its bizarre capitalisation to the fact that one Igor Daniloff is one of the progenitors of *Dialogue Science*. The Scheduler is obvious in its purpose.

The help files bear mention at this point as being admirable in their execution. Selecting help on any page or tab results in a representation of the page in question popping up in the help dialog. Selecting any part of this gives context sensitive help which is useful and concise. This is both necessary and well implemented both from a reviewer point of view and as a program without hard copy documentation.

DrWeb is, as will be seen to follow in the pattern already set, a very simple application to use with the minimum of operations – the main scanning area being the first seen and the more complex configuration options performed by means of drop-down menus. The major part of this screen is made up of the tree structure of those local areas available for scanning, with the results area below when the window is maximised or noteworthy files are present. To the right of the tree display is a button to refresh this part of the display, and a large button for the commencement of scanning.

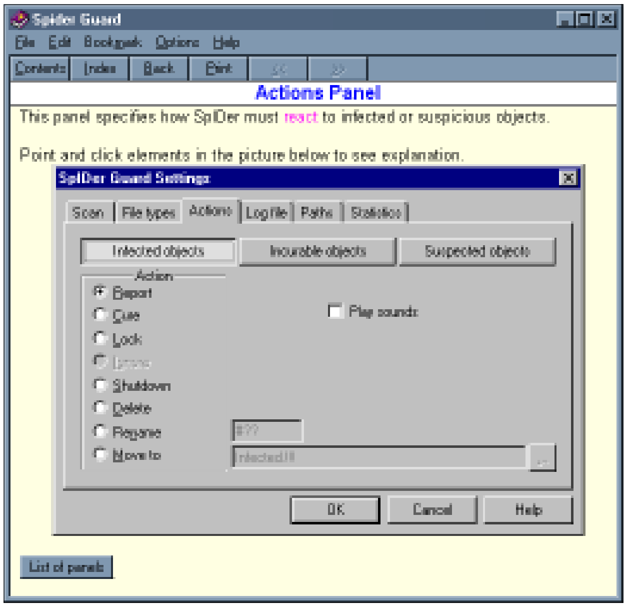


□ Show files in tree

Select drives
□ All floppy drives
□ All hard drives
□ All CD-ROM drives
□ All network drives

Settings
☑ Scan boot sectors
☑ Scan subdirectories

Dr.Web for Windows 95-2000 v4.22. Virus records: 23851 (23.02.2001)


Spider Guard
File Edit Bookmark Options Help
Contents Index Back Print << >>
SpIDer Guard Settings
Scan File types Actions Log file Paths Statistics
Infected objects Incurable objects Suspected objects
Action
● Report
○ Cure
○ Lock
○ Ignore
○ Shutdown
○ Delete
○ Rename
○ Move to
□ Play sounds
#??
infected.!!!
...
OK Cancel Help
List of panels

Actions Panel

This panel specifies how SpIDer must react to infected or suspicious objects.

Point and click elements in the picture below to see explanation.

To the left of the tree display are check boxes which select various areas for scanning, namely floppies, hard drives, CD-ROM or network drives. These are the more brute force methods of selection however, since individual areas can be selected by clicking upon an appropriate area of the tree. Also associated with the selection process are two more check boxes which determine whether boot sectors or subdirectories are included in the scanning process. Finally for this part of the GUI, a check box determines whether files are displayed in the tree structures.

These allow quick and easy access to the more basic functions of the scanner, while an icon bar and drop-down menus give access to the more complex parts of configuration. The icon bar, in fact, allows the entirety of the tree view to be replaced with a larger version of the report file or a set of statistics concerning the selected report. These statistics can be filtered by drive or taken as a whole. Also available in the menu is an option to clear the report list. As well as visual settings the menu bar has three extra icons available – settings, update and exit, the last simply quitting the program.

Updating via the Internet is initiated by a simple click of the icon, and despite the delay seen in connection on some occasions is very simple to perform. The setting icon gives access to the majority of configuration options. This is displayed as a tabbed set of areas falling under the categories of Scan, File Types, Actions, Log file, Paths, Events, Update and General.

‘Scan’ sets the defaults for both areas and methods of scanning. Default areas can be selected in line with the areas for scanning in the main screen, that is floppies, hard drives, CD-ROM or network drives. Of more interest is the scan methods area. This duplicates the settings for showing files in the scan tree and scanning boot sectors and subdirectories, but adds memory scans and heuristic analysis to the list of those selectable items. *DrWeb* has traditionally had very strong heuristics and relied upon these for a number of the viruses in the *VB* test-sets, though on occasion has suffered by dint of false positives in the Clean set as a result.

‘File Types Scanned’ is an area without shared commands from the main control area. It is set by default to scan files by format, which recognises those files likely to contain harmful code. Given past behaviour by the scanner this works with no problems. For more paranoid users ‘all files’ can be selected, or for those who have particular sets of extensions in mind two sets of extensions are selectable for scanning. One is designed for general use, the other for specific (e.g. .EXE only) files to be scanned without the need to disturb the settings of the other – not a detail that is likely to be used by many, but very nice to have for those occasions where a particular file type is suspect.

‘Actions’ is the home of the selections for what to do upon infection. Infected, Incurable and Suspicious objects are independently selectable, and the standard Report, Cure, Delete, Rename and Move To actions are all supported, with the quarantine area user definable. The report file is configured in the Log File tab, which can be appended or overwritten, and has a choice of character sets to be utilised, probably of more use in cyrillic language countries than in those using the standard western alphabet. The details to be included within the file can be fine-tuned, with scanned objects, file packer and archive details and statistics able to be independently chosen.

‘Paths’ is a less obvious control area, which determines where virus databases are searched for – allowing for central installations for updates, in addition to being the place where excluded folders are selected. ‘Events’ is a sparse area, selecting only whether sounds are generated when a virus is detected, and allowing a particular .WAV file to be selected. Needless to say, this was disabled in the tests as well as by default.

The ‘Update’ tab continues on the minimal theme – giving a URL for updates as its selectable object, and allowing user names and passwords to be supplied if this is not the standard Web site of *DialogueScience*. Last in this area comes the ‘General’ tab, giving control as to whether settings are saved on exit. For added control, window sizes or the like may be saved to the Registry, and the thread priority for scanning set at a higher or lower priority.

The *SpIDer Guard* portion of *DrWeb* is the on-access scanner and, again, the interface is by means of a tabbed box. Even more impressive is that the configuration changes available map those in the on-demand scanner, so that by knowing the interface for one, the interface to the other is easily understood. The tabs available here are Scan, File types, Actions, Log File, Paths and Statistics. Of these Log file, File Types and Paths are identical to the tabs seen in the on-demand scanner. The only fault that can be found with these settings is that for any configuration change it is necessary to reboot the machine.

‘Scan’ adds a selection to load at startup, and Virus Activity control activation. This latter is a further behaviour blocker which operates in addition to the heuristic analysis. Actions are slightly expanded from the on-demand page, allowing for the world record selection of actions seen in the *SpIDer Guard* help file screenshot on p.22. The final tab, ‘Statistics’, gives a breakdown of scanned, infected, suspicious, virus-like and modified virus files, and summarises what action was taken upon the discovery of each one.

Scanning Tests

The updated version of the virus definitions for *DrWeb* showed several differences in detection from the previous version. For one, the heuristic engine did not trigger at all – though this was not, in this case, a bad thing. All those viruses in the *VB* test-set which had been previously detected by the use of heuristics have in the last month been updated to be detected as specific viruses, which cannot be bad. In fact, all files were detected in every test-set other than a sprinkling of the Cryptor samples in the Polymorphic set. This was more worrying than it might otherwise be, since these files were detected in the previous Comparative.

Consultation with *DialogueScience* revealed they too had become aware of this new ‘feature’ and that it could be reversed by use of an older .DLL and would be changed in the next version of the product. When this older DLL was tried, sure enough, all the files in our test-set were exactly identified as viral.

Full detection is not of particular interest, despite being heartwarming for any developer, so additional tests were performed. The traditional test of ‘no heuristics’ would have been futile, and so a test without scanning engine was devised. *DrWeb* protests bitterly if the virus database files are removed, but by use of an empty database the heuristics could be tested alone, without interference. This test resulted in 14,299 suspicious files detected out of a total of 21,170. This figure does include a vast number of polymorphic viruses detected in large numbers, and so does not reflect a 67% detection rate by *VB* calculation protocols, but is nonetheless impressive. As might be expected, the more modern file viruses were less well-detected than the older, more family-based file viruses, and macro viruses were better detected than either of these.

In addition to the ‘no definitions’ test, another was performed, having been suggested at the VB2000 conference. This involves the testing of past versions of the virus engine against present versions of the *VB* test-set. The object is to emulate how a product will be able to detect viruses unknown at the time of its writing and hopefully project this into the future. This is far from a firm science, but eminently preferable to creating deliberate variants or novel viruses in the hope of determining what the heuristics of a product are capable of. Aside from the ethical considerations, the production of new viruses is more valid than mere ‘cut and paste’ on older viruses and far too time-consuming for any sane reviewer.

Diatribes aside, the product chosen for this testing was that from eighteen months ago, a long period indeed in terms of virus writing. In order to avoid the overall skewing of results too much by the presence of older viruses, the tests were performed only against the ItW set for the last Comparative – that being dated December 2000.

The one and a half year old product (October 1999) was version 4.14 and it performed, if not spectacularly well, adequately. The main problem, however, was that the majority of files detected as viral were exactly identified, having been in the WildList for this time.

Of the 838 files in the ItW set, 546 were detected exactly, while 22 were detected by heuristics. Of those undetected there were a large proportion of worms, perhaps not surprising given that these were barely a threat this time a year ago in comparison with today. Of those which were detected by heuristics all were Win32 file viruses; specifically W32/Kriz.4029 and .4050, W95/Lovesong.998, Win32/Funlove.4099 and W95/Spaces.1445. The latter only got into the WildList in December 2000 – so *DrWeb* performed admirably here. That none of those detected heuristically were macro viruses did come as something of a surprise.

Conclusion

DrWeb is a product which looks simple on the outside but is sufficiently complex under the skin to keep even the most frenzied tweaker happy. This exterior simplicity is a good example for some of the producers of the larger and more bloated products on the market to look at, especially when combined with stability.

With the detection rates displayed not open to criticism it would seem hard to find fault with very much at all in *DrWeb*. Searching hard, the need to reboot when reconfiguring does give some cause for complaint, though oddities revolving around this have been erased in this latest version. All in all, *DialogueScience* have reached an enviable position, and their main concern now will no doubt be to retain this as new threats emerge.

Technical Details

Product: *DialogueScience DrWeb for Windows 95–2000 v4.22.*

Developer: *DialogueScience Inc, 40 Vavilova Street, Moscow 117786, Russia*; Tel: +7 095 137 0150, email contact@dials.ru WWW; http://www.dials.ru/.

Price: Single PC – US$50, 10 PCs – US$230,
100 PCs – US$1040.

Test Environment: Three 750 MHz AMD Duron workstations with 64 MB RAM, 8 and 4 GB dual hard disks, CD-ROM, LS120 and 3.5-inch floppy, all running *Windows ME*. The workstations were rebuilt from image back-ups and the test-sets restored from CD after each test.

Virus Test-sets: Complete listings of the test-sets used are at http://www.virusbtn.com/Comparatives/NT/2000/11test_sets.html. A complete description of the results calculation protocol is at http://www.virusbtn.com/Comparatives/Win95/199801/protocol.html.

No responsibility is assumed by the Publisher for any injury and/or damage to persons or property as a matter of products liability, negligence or otherwise, or from any use or operation of any methods, products, instructions or ideas contained in the material herein.

SUBSCRIPTION RATES

Subscription price for 1 year (12 issues) including first-class/airmail delivery:

UK £195, Europe £225, International £245 (US$395)

Editorial enquiries, subscription enquiries, orders and payments:

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

Tel 01235 555139, International Tel +44 1235 555139
Fax 01235 531889, International Fax +44 1235 531889
Email: editorial@virusbtn.com
World Wide Web: http://www.virusbtn.com/

US subscriptions only:

VB, 50 Sth Audubon Road, Wakefield, MA 01880, USA

Tel (781) 2139066, Fax (781) 2139067

END NOTES AND NEWS

InfoSec 2001, Europe's largest IT security event, is to take place from 24–26 April 2001 in the National Hall, Olympia, London, UK. See the Web site http://www.infosec.co.uk, or find out more about the event by emailing infosecurity@reedexpo.co.uk.

***Symantec* announces the establishment of a new division in the company – the Service Provider Solutions Division –** which is to create products targeted at Internet and application service providers, portals and global Internet carriers. *Symantec* aims to embed its AV products throughout the Internet. For more information visit the Web site; http://www.symantec.com/.

iSEC Asia 2001 is to be held at the Singapore International Convention and Exhibition Centre from 25–27 April 2001. The conference and exhibition covers IT security topics from anti-virus through encryption to biometrics and digital signatures. For more information and a booking form contact Stella Tan; Tel +65 322 2756 or email stella@aic-asia.com.

***F-Secure* has released *F-Secure Anti-Virus for Internet Mail* and *F-Secure SSH Server 5.0 for Windows*.** The former product supports *Windows NT 4.0* and *Windows 2000* and the email server can sit on any platform. In an unrelated announcement, *F-Secure* plans the integration of its products with the Enterprise Management Systems (EMS) from *Computer Associates* and *BMC Software*. See the Web site http://F-Secure.com for further details.

InfoSec Paris 2001, the 15th information systems and communications security exhibition and conference, will take place at CNIT, Paris-La Défense, France from 29–31 May 2001. Companies wishing to participate in the exhibition are encouraged to contact the organisers; Tel +33 0144 537220, or email salons@mci-salons.fr.

Norman ASA* has released *Norman Personal Firewall 1.0 for office workstations and home PCs. It supports *Windows 95/98/ME* and *NT/2000 Professional*. Available for download from http://norman.com/.

iSEC Australia will take place in Halls 5 & 6 of the Sydney Convention & Exhibition Centre from 6–8 August 2001. For information on how you can be a sponsor, exhibitor or delegate, visit the Web site http://www.isecworldwide.com/isec_aus2001/. Alternatively contact Chris Rodrigues; Tel +61 2 9210 5756.

Elron Software* has agreed to integrate *McAfee* anti-virus technology into its *Internet Manager Anti-Virus Solution, for use alongside *Internet Manager Message Inspector* and *Internet Manager Web Inspector*. See http://www.elronsoftware.com for more details.

Central Command* has announced the availability of *AVX Scan Online, on-demand virus protection through your Internet browser. It supports *Windows 95/98/ME*, *NT/2000* or *Linux* and *Internet Explorer v4.0* or *Netscape 4.x*. For more details about this free downloadable product visit the Web site http://www.avx.com/.

***GeCAD Software* is launching two new products at CeBIT in Hannover, Germany from 22–28 March.** *RAV Enterprise* and *RAV AntiVirus Desktop v8.2* will be available to the public then. For more information see the Web site http://www.ravantivirus.com/.

***Sybari Software* announces the release of *Antigen 6 for Exchange 2000 and 5.5*.** The product offers multiple anti-virus scan engine support from *NAI*, *Norman*, *Sophos*, *CA Vet* and *CA InoculateIT*. See the Web site http://sybari.com for more details.

Linux Expo 2001 Exhibition & Conference is to take place at Olympia, London in the UK from 4–7 July 2001. To find out about exhibition opportunities or to register for the show, email the organisers jonathan.neastie@itevents.co.uk or visit the conference Web site http://itevents.co.uk/.

***Sophos* is to host a two-day workshop entitled 'Investigating Computer Crime & Misuse' on 10 and 11 April 2001** at its training suite in Abingdon, Oxfordshire, UK. For details about the different courses and training days available, or to reserve your place, contact Daniel Trotman; Tel +44 1235 559933, or email courses@sophos.com.