ISSN 0956-9979 SEPTEMBER 2002

VIRUS BULLETIN

THE INTERNATIONAL PUBLICATION ON COMPUTER VIRUS PREVENTION, RECOGNITION AND REMOVAL

IN THIS ISSUE:

• **Where there's a will …** *Eset Software*'s Anton Zajac vents his feelings about certain anti-virus testing procedures on p.2.

• **What a gem:** The author of W32/Gemini has produced a series of 'one-of-a-kind' viruses; Peter Ferrie investigates the latest in the collection of unusual techniques on p.4.

• **Open to question:** *VB* talks to two of the developers behind the OpenAntiVirus Project about the goals of the team, the ongoing projects and its reception in the AV community. See p.6.

• **GFI Friday!** Matt Ham puts *GFI MailSecurity* through its paces and reports back on p.18.

CONTENTS

COMMENT

We Will Find the Way ...

When Hannibal led his troops through the freezing peaks of the snow-covered Alps, he declared: 'We will find the way … and if not we will make one!'. The fatalism and determination of Hannibal's famous statement has inspired many and will continue to do so. However, when a frozen death is not the unspoken alternative, I believe that we must take care to ensure a noble cause is not used to justify unethical behaviour.

When I read consumer computing and technology publication *CNet*'s latest online reviews of a number of anti-virus products (http://www.cnet.com/), I realized the reviewers had fallen into 'Hannibal's trap' of obsessing over a chosen path despite its obvious folly. The goal was to provide readers with valid information on the quality of the products being tested. They failed – the question is why.

"The reviewers had fallen into 'Hannibal's trap' of obsessing over a chosen path despite its obvious folly."

To the less well informed reader, this comparative review of anti-virus products appears to be based upon a solid, perhaps even scientific, testing methodology. To test the detection rate of the products, the reviewers used 'synthesized viruses' generated by the Rosenthal Utilities (RU). However, the use of RU was discredited by the AV community several years ago. Almost two years ago, well-known anti-virus researcher Joe Wells wrote an open letter to *CNet*, pointing out the flaws in their testing methodology. The letter was co-signed by 19 prominent members of the anti-virus community (see *VB*, November 2000, p.3). However, *CNet* continued to carry out their tests using RU and reporting consequently flawed results on their website. Perhaps they honestly thought the industry experts were wrong; perhaps they thought anti-virus products should detect RU generated synthesized viruses. Perhaps they were just obsessed with not admitting an error.

I contacted *CNet*'s Chief Editor but received no answer to my question regarding *CNet*'s definition of a computer virus, nor to the question: 'Is the attachment [generated by RU] a virus?' which I posed specifically to *CNet*'s Technical Editor and to the CEO. However, I was assured repeatedly that *CNet* was 'phasing out further use of the RU for anti-virus testing'. I could accept *CNet*'s diplomatic silence more easily had the flawed test results been removed from *CNet*'s websites and an appropriate apology been made to all affected parties (including their misled readers).

Without an answer, my search for the truth continued (I had to 'find the way'). Analysis of all the simulated COM and EXE files revealed a striking simplicity of all (roughly 2000) generated RU samples. All the samples have similar structure and one thing in common: none of them is a virus nor a virus-infected file. A virus alarm triggered by any of the RU samples is a false alarm.

RU are distributed with documentation defining their scope of applicability, which states 'the simulators all produce safe and controlled dummy test virus samples … these samples contain the signatures (only) from real viruses. The programs themselves are not really infected with anything … The simulators' ability to actually test products is limited.' The documentation goes on to state: 'These test virus simulators are not intended to replace the comprehensive collection of real virus samples.' In spite of the striking clarity of RU's disclosure, *CNet* used samples 'not really infected with anything' to perform tests whose results were to be presented to public as real and relevant – in my view this was irresponsible.

The laws of quantum mechanics have never ceased to amaze me. When a test is performed on a set of identically prepared systems (e.g. electrons) the test results differ as determined by the probabilistic nature of those laws. *CNet*'s 'laws' for testing anti-virus products are more peculiar still. Regardless of what set of products is tested, and regardless of what feature is tested, the outcome is always the same: there is only one perfect product. *CNet* will always 'find its way'.

Perhaps times are changing – it took *CNet* two years to discontinue its flawed testing; we can only guess the time it will take *CNet* to stop misleading its readers.

Dr Anton Zajac, VP, Eset Software

NEWS

Bring on the DEET

This month the award for the most tenuous product-pushing story goes to *BitDefender*, whose marketeers have seen fit to warn the world about the vague possibility that our computer systems might fall victim to 'the world's most publicized subject today'. (No, not violence in the Gaza Strip, nor flooding in central Europe and China, nor the Kashmir dispute.) According to *BitDefender*, the West Nile virus might be about to spread to our computer systems: Mihai Radu, Communications Manager at *BitDefender* warns, 'The mosquito-borne disease could easily become a computer infection.' [*Note*, VB *does not recommend application of DEET to your hard drive*.]

In fact, *BD*'s argument is that, because 'West Nile virus' has become one of the top search terms on the Internet, there is a significant likelihood that virus writers will exploit the high level of interest in the subject. Mihai Radu says 'Experience [has] proved that any subject important enough to represent a story for *CNN* or other important media at national or international level, triggers a quick-spreading computer virus, conjectured by that story.' So, just to clarify, that's *any* subject reported by *CNN* or other international news organizations will trigger a computer virus … Happily, we may rest assured that *BitDefender*'s experts are working on a solution to protect users against potential malicious code related to the West Nile virus. Let's hope they have their eye on *CNN* for all the other breaking news stories ▮

Virtually There

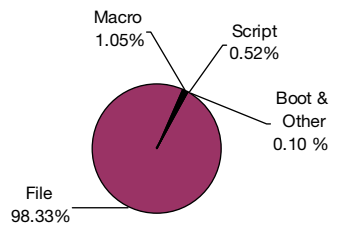
The Infosecurity show and exhibition has gone virtual with the launch of the first Infosecurity World Online exhibition. Following registration on the website, the visitor is welcomed into a very peaceful looking virtual convention centre with a scattering of smartly dressed delegates (no shorts-and-t-shirt-wearing throng at this show). The site includes virtual exhibition stands, a virtual newsroom (containing case studies, white papers and news), and a virtual seminar room where flash presentations or keynote speeches can be played. *Trend Micro*, *Clearswift*, *Sophos*, *Compaq*, *Pentasafe* and *Computer Weekly* are amongst those currently 'exhibiting' at the show. Their stands enable visitors to gather details about products and services, adding the relevant information to a virtual exhibition bag, the contents of which will be forwarded by email. Requests may be left for the exhibitor to contact the visitor in person, and press releases and links to the exhibitors' websites are provided – in fact pretty much everything you would get at a 'live' exhibition. *VB* notes one glaring omission though: what about the free coffee, sweets and merchandising goodies? See http://www.infosecurityworldonline.com/ ▮

Prevalence Table – July 2002

Virus	Type	Incidents	Reports
Win32/Klez	File	4833	57.65%
Win32/Frethem	File	1041	12.42%
Win32/Yaha	File	792	9.45%
Win32/SirCam	File	701	8.36%
Win32/Magistr	File	396	4.72%
Win32/BadTrans	File	100	1.19%
Win32/Nimda	File	82	0.98%
Win32/Hybris	File	63	0.75%
Win32/Higuy	File	55	0.66%
Laroux	Macro	49	0.58%
Win95/CIH	File	29	0.35%
Win32/Elkern	File	21	0.25%
Win32/MTX	File	21	0.25%
Win32/Gokar	File	18	0.21%
Win32/Fbound	File	16	0.19%
Win32/Funlove	File	12	0.14%
Kak	Script	11	0.13%
Win32/Gibe	File	10	0.12%
LoveLetter	Script	9	0.11%
Divi	Macro	8	0.10%
Haptime	Script	8	0.10%
NoClose	Script	6	0.07%
AntiCMOS	Boot	5	0.06%
Redlof	Script	5	0.06%
Win32/Datom	File	5	0.06%
Win32/Nahata	File	5	0.06%
Win32/Onamu	File	5	0.06%
Win32/QAZ	File	5	0.06%
Others [1]		72	0.86%
Total		8383	100%

[1] The Prevalence Table includes a total of 72 reports across 38 further viruses. Readers are reminded that a complete listing is posted at http://www.virusbtn.com/Prevalence/.

Distribution of virus types in reports



VIRUS ANALYSIS

Attack of the Clones

Peter Ferrie
Symantec Security Response, Australia

Once again, old ideas have been given a new lease of life on the *Windows* platform. The idea used by W32/Gemini is, perhaps, all the more interesting because it came not from the era of MS-DOS and its variants, but from an operating system that existed a decade earlier. The author of Gemini has produced a series of 'one-of-a-kind' viruses; whatever the motive behind it, this is another in the collection of unusual techniques.

All the Merry Men

According to the background information, the first demonstration of this technique was seen in the mid-1970s on the Xerox CP-V timesharing system. It was described more recently by Péter Ször in his presentation at the *Virus Bulletin* conference in 1999, when he named the idea 'The Twins'.

The idea is that two co-dependent processes are run simultaneously, each one checking the state of the other as it runs. In the case of the Xerox CP-V implementation, each of those processes would, among other things, send console messages to the other process, the text being directed from 'Robin Hood' to 'Friar Tuck', or from 'Friar Tuck' to 'Robin Hood'.

That version elevated its privileges to supervisor level and then proceeded to interfere with the system by dismounting tapes and walking drives. W32/Gemini does not demonstrate any of these characteristics, remaining as a user mode application and having no noticeable effects, but what it does share is the active prevention of the termination of the malicious processes.

Anti-anti-anti-anti …

When Gemini is started for the first time, it decompresses and drops a standalone executable file that contains only the virus code, using a 'fixed' (taking into account the variable name of the *Windows* directory) filename and directory.

An occasional technique against such viruses is to create a directory or read-only file with the same fixed name. Unfortunately, Gemini contains code to work around both of these cases. If such a file exists there, then its read-only attribute (if any) will be removed, and the file will be deleted. If a directory exists there instead, then it will be renamed to a random name.

The decompressed file has an unusual structure – the MZ header, PE header, and the section table are overlapped and truncated. At a glance, it appears to be a file with a corrupted structure, perhaps as an anti-heuristic method, since corrupted files might not be scanned by some anti-virus software. Despite this apparent corruption, the file runs correctly on all current *Windows* platforms (*95*, *98*, *ME*, *NT*, *2000*, *XP*). This technique has been used by several viruses in the family. To create such a file would require a significant amount of trial and error … it seems that someone has a lot of time on his or her hands.

Are You Talking to Me?

Once the file has been dropped, it is executed. The code in the dropped file calculates the checksum of itself and stores it in a fixed location in the code. After the checksum has been stored, the virus creates two events with random names, one for each copy of the code that will eventually be running. These events are created in an unsignalled state, and requiring manual reset.

Then the virus runs a second copy of itself, passing these event names and the process identification number of the first process as command-line parameters. When the second process is created successfully, *Windows* returns to the first process the process identification number of the second process. This is how the virus establishes its inter-process communication.

The Eternal Cycle

In order to understand how the communication works, it is perhaps easiest to designate one process 'active' and the other process 'passive', where the process that has control at any instance is considered to be the 'active' process, and the other process is 'passive'. Note, however, that the designation is arbitrary because the control will be swapped continually between the two processes.

The active process checks the state of the event belonging to the active process. If the event in the active process has been reset by the passive process, then the event in the passive process will be reset by the active process. The code in the passive process will then be checksummed to detect tampering. The checksum algorithm is simply a sum of the bytes in the code. If the checksum matches the expected value, then the active process will set the event to a signalled state in the active process and wait for a short time for the event to signal in the passive process. If the event signals in time in the passive process, then the code will begin the checks again.

Rise Like the Phoenix

Thus, if the event in the active process was not reset by the passive process, then the virus considers that the passive

process has been suspended or terminated, and will run another copy of it. However, if the code changes in the passive process, the virus will behave as though it had never been run before on that computer, and will start two copies of the code, with new event names and process identification numbers.

As *Windows* switches from one process to the other, so the currently active process becomes the passive process, and the currently passive process becomes the active process. The only way to disable the processes is to terminate both of them simultaneously (at least, within the period that the active process waits for the event to signal in the passive process); however this can be a difficult task to achieve in *Windows*.

Meanwhile, Back at the Ranch

In order to perform its actions, the virus uses the Structured Exception Handler list to gain access to KERNEL32.DLL. After gaining access to KERNEL32.DLL, the virus will retrieve the addresses of API functions that it requires, using the increasingly common CRC method to match the names.

Unlike the authors of some of the other viruses that use the CRC method, the author of this virus was aware that APIs must be stored in alphabetical order, so there is no need to search the CRC table repeatedly. Additionally, the virus has support for both ANSI and Unicode functions merged into a single routine, and selects the set of APIs that is appropriate to the current platform (ANSI for *Windows 9x* and *ME*; Unicode for *Windows NT*, *2000*, and *XP*).

In addition to the process synchronization, Gemini will create a thread that searches for files in all subdirectories on all fixed and mapped network drives. This thread is executed every ten minutes, and is run by all running copies of the code.

The file-searching algorithm is identical to the one used by the other viruses in this family, using a linked-list instead of a recursive function. This is important from the point of view of the virus author, because the virus will infect DLLs, whose stack size can be very small.

Filters

Files are examined for their potential to be infected, regardless of their suffix, and will be infected if they pass a very strict set of filters. The first of these filters is the support for the System File Checker that exists in *Windows 98/ME/2000/XP*.

The virus author was aware of the fact that the IsFileProtected() API requires a Unicode path, while directory searching on *Windows 9x* and *ME* require an ANSI path, so the virus transforms the path dynamically.

The remaining filters include the condition that the file being examined must be a character mode or GUI application for the Intel 386+ CPU, that the file must have no digital certificates, and that it must have no bytes outside of the image.

Touch and Go

When a file is found that meets the infection criteria, it will be infected. If relocation data exist at the end of the file, then the virus will move the data to a larger offset in the file, and place its code in the gap that has been created. If there are no relocation data at the end of the file, then the virus code will be placed here. The entry point is altered to point directly to the virus code.

Once the infection is complete, the virus will calculate a new file checksum, if one existed previously, then continue to search for files.

Once the file searching has finished, the virus will allow the application to exit by forcing an exception to occur. This technique appears a number of times in the virus code, and is an elegant way to reduce the code size, in addition to functioning as an effective anti-debugging method.

Since the virus has protected itself against errors by installing a Structured Exception Handler, the simulation of an error condition results in the execution of a common block of code to exit a routine. This avoids the need for separate handlers for successful and unsuccessful code completion.

Conclusion

While W32/Gemini offers nothing new in terms of its replication and infection methods, it does take a step forward in the implementation of 'anti-anti-virus' techniques.

As operating systems increasingly hide or remove the ability to produce and use a bootable floppy disk, so users rely increasingly on anti-virus software being run on a computer that is actively infected. The initial response to this was memory scanning and process suspension, termination, or alteration. Now we need a new response. We play the game but the rules keep changing.

W32/Gemini	
Aliases:	W32/Chiton.
Type:	Memory-resident parasitic appender/inserter.
Infects:	*Windows* Portable Executable files.
Payload:	Actively prevents viral process termination.
Removal:	Delete infected files and restore them from backup.

INTERVIEW

Open All Hours

The OpenAntiVirus Project was started two years ago to address what was seen by its founders as a serious lack of open-source solutions in the anti-virus field. The intention was to build and nurture a network of AV developers within the open source community by providing the relevant resources for communication and project management. VB *spoke to co-founder Rainer Link and developer Kurt Huwig about the project.*

How did the project come about?

The OpenAntiVirus Project was officially started in August 2000, when we registered the project at sourceforge.net and we reserved the openantivirus.org domain.

It was started by Howard Fuhs and Rainer Link – the idea itself was conceived in January 2000 during a telephone call between the two of us. We both have a very strong belief in open-source software and we feel that, especially in the field of security, open protocols and open/free software are essential.

OpenAntiVirus is a platform for those who are seriously interested in anti-virus research and network/computer security. Its aim is to facilitate communication amongst researchers and consequently the development of solutions for various security problems and development of new security technologies. Researchers are encouraged to work together, share their ideas and re-use existing code. Although a healthy amount of competition is welcomed, all-out 'war' between projects is discouraged!

Mailing lists are provided as the main source of communication for participants. Currently there are three lists (each provided by a sourceforge.net version of GNU Mailman): openantivirus-announce is a mailing list dedicated to announcements from the OpenAntiVirus team; openantivirus-discuss is a general discussion mailing list; and openantivirus-developer is for developers.

The project is open to anyone who has a genuine and serious interest in white hat anti-virus research and computer security.

What are the ongoing projects?

There are a number of official OpenAntiVirus.org (OAV) projects. ScannerDaemon, and VirusHammer are the first implementations of a GPL'ed virus scanning engine (VirusHammer is a standalone virus scanner that can be run by end users); squid-vscan is a third-party application which allows traffic passing through Squid HTTP-proxy to be scanned for known viruses; samba-vscan is a proof-of-concept module for Samba, using the virtual file system features of Samba 2.2.x/3.0 alphaX (this also supports a wide range of commercial anti-virus scanners); finally, a Mini-FAQ text file is maintained, which lists anti-virus products available for Unix/*Linux*.

Alongside the official OAV projects, the OpenAntiVirus.org members have developed a number of applications and tools. Email virus scanner AMaViS was initially set up by Christian Bricart and is maintained by Lars Hecking and co-developed by Rainer Link; httpf is a WWW security proxy co-developed by Gregor Goldbach; and Inflex and XaMime are email content-filtering and virus scanning tools developed by Paul L. Daniels, whose SignatureDB provides signatures/fingerprints of common non-viral but undesirable emails or files.

Many of the subscribers to our mailing lists are also working on their own projects.

How many developers work on the project? Is there a central core of developers?

The core OpenAntiVirus team consists of co-founders Rainer Link (project admin) and Howard Fuhs, webmaster Frank Ziemann and developers Christian Bricart and Kurt Huwig.

Currently, most of the development work is carried out by Kurt Huwig and Rainer Link. Kurt's work is focused on the core virus scanning engine (ScannerDaemon/VirusHammer) and the third-party application squid-vscan. Rainer's work is focused mainly on developing third-party applications such as AMaViS and samba-vscan.

Your primary scanning engine is written in Java – what influenced this choice? Do you think the speed of your scanning engine suffers because of it?

(*Kurt*) I knew this would be one of the top FAQs even before I released the first version. Because everyone uses the poor Java implementations in browsers, they think Java is inherently slow.

The fact is that the current engine scans about 12 Megabytes per second on a Duron 800, which is more than a 100 MBit NIC can transfer, so I do not think that it presents a problem for anyone's Internet connection besides the real big carriers.

I carried out some testing before I started to develop the scanner and these tests showed that an optimized version can scan 125(+) Megabytes per second on an Athlon 600 (I switched machines in between). This is about the memory transfer rate of my current machine, so I do not think that speed will be an issue for me in the future.

Another developer has implemented a scanner based on the signature file of the OAV project. The following is a comparison of scanning speeds between the OAV Java implementation (ScannerDaemon) and ClamAV, which is written in C:

100 Mb random file	
ScannerDaemon 0.5.1	8.95 s
ClamAV 0.20	9.012 s
5 Mb random file	
ScannerDaemon 0.5.1	0.22 s
ClamAV 0.20	0.468 s

The differences are minimal – clamscan builds the tree each time. (This was tested on a 1.2 GHz Athlon desktop, with JRE 1.4.0 and the same virus database and algorithm settings.)

Do you find you get a large number of patches/suggestions from users?

We do receive some suggestions from users and also some patches. The core has yet to settle and I guess once it is stable, people will write extensions to it.

How much interest has the project generated? Do you have a rough estimate of the number of users?

The openantivirus-discuss mailing list has roughly 300 subscribers. Recently the OAV ScannerDaemon was added to Debian and we were asked not to change the interfaces for the virus updates. According to the peak of downloads after a new release of the signature file, I would estimate that we have somewhere between 400 and 1000 active users.

Primarily, who do you find your users are?

(*Rainer*) I'd say mostly sysadmins of smaller companies, but it's difficult to give exact figures here. *SuSE* has shipped AMaViS since *SuSE Linux 7.2* (and offers it for users of the *SuSE* email server); samba-vscan is shipped on the update release CD of the *SuSE Linux Enterprise Server 7* and it will be shipped on the next *SuSE Linux* release, too.

(*Kurt*) The majority of feedback I get is from sysadmins trying to get OAV into their system.

What are the benefits of open-source anti-virus software? Do you feel there are downsides?

We are not tied to any business plans or political restrictions. We can detect dialers and governmental tapping software, which some of the commercial vendors choose not to do.

You can use our products in other open-source products without the need to buy a licence either for yourself or for your customer. Recently, we have been contacted by a group that writes an IRC client and who would like to add virus scanning to their product for the file transfer.

Open source prohibits the use of NDA documentation, but we can use a number of GPL-licensed algorithms so, altogether, it is easier to do the open-source thing.

Do you think the availability of commercial scanning libraries on* Linux *and* FreeBSD *inhibits the adoption of open-source anti-virus software?

Currently, there is not much competition between commercial products and our product. The commercial products detect many more viruses, so anyone who relies on virus detection for their protection really has to choose the commercial scanners.

On the other hand, our product is free of charge, which makes it ideal for use in universities, schools, NGO, the health care sector, projects in the developing world and so on – organizations and projects that may not have sufficient funds to pay for the commercial product licences. Using our product they get at least some protection against some 95 per cent (plus) of all viruses.

A significant barrier to entry into the anti-virus field is getting hold of real virus specimens – how do you go about attaining them?

(*Kurt*) The anti-virus community is quite friendly to me. Most of the time when I ask for a virus sample, someone sends it to me. Currently, I have more samples than I can analyse, so I am experiencing something of a bottleneck.

I do not have access to the big 'In the Wild' collections. However, at the current time my primary concern is to stabilize the engine and once that has been achieved I will head towards the integration of more viruses.

Have you encountered any hostility from the anti-virus industry?

Not via personal mail, but there has been some negative feedback on some newsgroups/mailing lists.

(*Rainer*) I have a very good relationship with a lot of anti-virus vendors, especially since some of the companies benefit from projects like AMaViS or samba-vscan – I know that some anti-virus companies who do not have their own email gateway solution suggest AMaViS to their customers.

Does OAV plan to undertake any specific projects in the future?

There are a number of projects in the pipeline for the OpenAntiVirus team, including a rescue disk/CD, general on-access scanning and a remote management system.

For more information on the OpenAntiVirus project, including details of how you can join the mailing lists and contribute, visit http://www.openantivirus.org/.

TUTORIAL

Mission Impossible – Part 2

Aleksander Czarnowski
AVET Information and Network Security, Poland

Last month (see *VB*, August 2002, p.10) we left our 'Secure IIS' mission having created a secure *Windows 2000 Server* installation with fully functional *IIS*. This month we take up where we left off and look at securing *IIS* services as well as making use of a couple of *Microsoft* security tools.

Hfnetchk and MBSA

The *Microsoft* tools hfnetchk and MBSA are intended to help in securing the *Windows* installation process. Hfnetchk is a hot-fix/Service Pack checking tool. When run, it checks the Registry and files against a database of available hot-fixes and reports any missing patches. Microsoft Baseline Security Analyzer (MBSA) is a more advanced tool. It uses hfnetchk to identify missing patches, but also searches for vulnerabilities in the *Windows* system installation.

While the ideas behind these tools seem flawless, their inner workings are far from such. First, hfnetchk uses *Internet Explorer* to retrieve a hot-fix database file in a CAB format from the *Microsoft* website. If an attacker is able to sabotage DNS servers (for example with dsniff dnsspoof, as in the recent exploitation of *IE* SSL vulnerability), he can redirect *IE* to any site and try to exploit an *IE* vulnerability. *IE* vulnerabilities are very common and *Microsoft* does not always release hot-fixes immediately after their discovery. Fortunately, hfnetchk can be supplied with a local CAB file. MBSA would be a useful tool if it were used to scan the *Windows* system remotely. Both MBSA and hfnetchk do have such an option – however, to scan your server successfully you need to reconfigure it in such a way that the security level will be lowered. The moral of this story is simple: use both tools only locally and with a mssecure.cab file supplied locally.

Securing IIS

One of the first steps in securing *IIS* should be the removal of all sample applications and directories. Some of these are located in the Inetpub structure. The following is a list of directories from which content should be removed:

- \Inetpub\iissamples (sample files)
- \Inetpub\AdminScripts (administration scripts)
- %systemroot%\help\iishelp (*IIS* documentation)
- \Program Files\common files\system\msadc (data access)

Note that *IIS* documentation is not included in *Windows* help. So, instead of deleting the directory, you can simply move it to a different partition outside the Inetpub structure. After the deployment process you should delete it.

The next step is to remove all unnecessary and potentially dangerous script mappings:

Script type	Mapping
Web-based password reset	.htr
Internet database connector	.idc
Server-side includes	.stm .shtml .shtm
Internet printing	.printer
Index server	.ida .idq .hta

This can be done through the Home Directory Tab, using the Configure button in the Default Web Site Properties window. If you are wondering why you should do this, cast your mind back to the ida or .htr buffer overflows (*MS Security Bulletins 01-033* and *02-018*).

Now it's time to set ACLs (Access Control Lists) on files. The default *Windows 2000* and *IIS* installations have weak ACL settings. Don't forget that, before you set up ACLs for *IIS* files, you must set ACLs for *Windows* (see http://www.microsoft.com/technet/security/tools/chklist/w2ksvrcl.asp). *Microsoft* recommends the following ACL settings for *IIS* files:

File type	Description	ACL
htm, html, txt, gif, jpg	static web files	Everyone: read Administrators: full control System: full control
inc, shtm, shtml	Include files	Everyone: execute Administrators: full control System: full control
asp	Scrip files	Everyone: execute Administrators: full control System: full control
exe, dll, cmd, pl	CGI scripts	Everyone: execute Administrators: full control System: full control

Ideally, you should never provide any exe, dll or cmd files on your web server. You should also check every CGI and ASP file you post very carefully. In many cases those files can provide a dangerous mechanism for remote users.

We also need to take care of ftproot (the default *IIS* ftp home directory) and mailroot ACLs. In the default installation, full control is assigned to the 'Everyone' group. While this makes it very easy to deploy the ftp server, it is not the best setting possible. The appropriate settings differ from site to site, depending on the role of the *IIS* server.

If you are running an ftp server you should not allow web access, and vice versa, so that if there were to be a vulnerability in the web service no one would be able to compromise the ftp server with it. Neither should you allow the uploading of files via ftp, since ftp servers with writeable directories that can be accessed anonymously can become repositories for warez or porn. Every visit should be logged.

Some sources suggest that the IIS metabase should be renamed. I don't advise this action, as the *IIS* metabase is usually accessed through API, so the file name does not play any role here. However, it is possible to imagine a very simple backup script that would relay the default name to the backup *IIS* metabase, for example. Changing the file name would only break the script without increasing the security level. There is one good rule for every hardening process: don't go too far. After a certain point you can only break more things than you secure. Security is a balance between protection mechanisms and usefulness.

Urlscan and IIS Lockdown

Microsoft provides Urlscan and IIS Lockdown free of charge, and every publicly accessible *IIS* server should run these tools.

There are a few versions of Urlscan available, the most recent being 2.5. This version must be installed over Urlscan 1.0 or 2.x. To do this you must download IIS Lockdown first. After successful downloading you must run it to extract urlscan.exe:

```
iislockd.exe /q /c /t:c:\ld_files
```

This will place Urlscan.exe in the c:\ld_files directory. Install it. Now you can download Urlscan 2.5 from the *Microsoft* website. At the command prompt type:

```
urlscan.exe /x
```

This will extract the Urlscan.dll file. Urlscan 2.5 can restrict request size so, effectively, it can protect against future buffer overflows. It can also log long URLs.

Urlscan-SRP is a version of Urlscan which protects against the vulnerabilities listed in *MS 02-018*. The difference between Urlscan and Urlscan-SRP is the way in which they handle chunked encoding transfers. Urlscan-SRP blocks all chunked encoding transfers and uploads are limited to 30 MB.

IIS Lockdown is a *Microsoft* wizard that will allow you to configure Urlscan and *IIS* services easily.

IIS and ISA Server

Remember your mission briefing? You were instructed to use only *Microsoft* products or open-source projects (this exception was made specifically for OpenSSH and snort, should you wish to run a NIDS on your *IIS* server). Now it's time for network traffic filtering. If you have ever looked at the options provided by network and dial-up connections, you probably found IP packet filtering options. Unfortunately, *Windows 2000* filtering options are not very flexible. But if this is the only option, you should block all TCP and UDP ports on external interfaces with the following exceptions:

- 21 TCP if you are providing ftp server access
- 80 TCP if you are providing web server access
- 443 TCP if you are providing SSL web server access
- any other ports if you are providing additional servers or changed default IIS ports.

However, you should always place your server behind a firewall. *Microsoft* provides such a product: *ISA Server 2000*. There is not enough space in this article to describe the *ISA Server* hardening and configuration process, so I will provide only a few tips. First, apply SP1 (or newer) to *ISA*. Also, remember that *ISA Server* shouldn't be run on *IIS Server*. It is possible to run both *IIS* and *ISA* on the same host at the same time. You just need to change the default *IIS* web TCP port from 80 to something else so you would be able to administer *ISA* and access *IIS*. Such a setup is not recommended in production environments.

Until this point we have been doing everything to limit possible attacks. Now we will concentrate on worms. *Microsoft* has published advisories regarding Nimda and Code Red in conjunction with *ISA Sever 2000*. You should read both documents (see http://www.microsoft.com/) even if you are not using *ISA* – some rules are so general that they can be applied to any other packet filter and proxy-based firewall solution. Here is a short list of possible settings:

- Block any tftp (and ftp if not needed) traffic
- Block all NetBios traffic
- Filter SMTP traffic and deny possibly dangerous attachments (you should never install *Outlook* on your *IIS* server).

Game Over?

Congratulations. You have successfully deployed a secure *IIS* server. But if you think that this is the end of the mission, think again. We have only minimized the risk of successful attack. Now you must prepare to install new hotfixes (there will be at least a few new ones by the time this article is published) and Service Packs. You also need to monitor system and *IIS* logs. If you installed snort you will need to spend some time configuring it and modifying its signatures for your needs (note: snort does not run on multiprocessor machines because WinPcap drivers don't support such architecture).

Oh, and did you know that it was possible to secure *Linux*-based *Apache* installations against the chunked encoding exploit before we even knew about this vulnerability? But that is a subject for another mission …

FEATURE 1

Trends in Malicious Internet Traffic

Andre Post
Symantec Security Response, The Netherlands

The Internet is becoming increasingly polluted by malicious software travelling from one computer to another. This malicious traffic is caused by worms, hackers scanning for unpatched vulnerabilities, hackers using backdoor Trojan horses, and various other sources of digital mischief.

Between May 2001 and March 2002, *Symantec* asked a number of volunteers to submit their *Symantec Desktop Firewall* logs to the company in order to help determine the current risk associated with being connected to the Internet. The volunteers were from Belgium, Denmark, Finland, France, Norway, the Netherlands, Sweden and the UK.

The log files provide a rich source of information about the nature of Internet traffic over the period of time during which they were recorded. Using this data, we can observe interesting trends in malicious Internet traffic. For example: how often do HTTP worms like CodeRed and Nimda present themselves to the average user? Is there a regional difference in hacker preference for backdoor Trojans or is SubSeven every hacker's number one?

Malicious Internet Traffic

Over the past decade, the Internet has evolved to become the largest computer network in existence. It facilitates vast amounts of data exchange between many different computers worldwide. The technical nature of such data exchange can vary greatly, e.g. requesting the content of a website, sending an email message, file-sharing over a peer-to-peer network, and so on.

Initially, the Internet was used solely as designed: to provide unprecedented volumes and speeds of information exchange. But soon it was discovered that the Internet would facilitate the propagation of malicious software. The days of regionally-contained virus epidemics were numbered.

Virus writers and hackers discovered the means to create worms and other malicious software that use the Internet to propagate or to gain access to many computers in a relatively short time frame. Since the appearance of W97M.Melissa@mm in 1999, malicious use of Internet bandwidth has increased tremendously. It is not only worms that pollute the Internet in this way, but also backdoor Trojans and other software with a malicious payload. Worm outbreaks, successful hack attempts, and denial of service attacks frequently make the news headlines.

An increasing number of individuals and businesses rely on the Internet to perform financial transactions and other operations involving confidential data. This indicates clearly that it has become even more important to protect sensitive data from hackers. Nowadays, there is a huge ongoing effort by hackers to try to gain unauthorized access to systems in all levels of today's digital society.

In this article, two types of Internet traffic are discussed. These are 'Malicious Internet traffic', which is considered to be Internet traffic that contains viruses, worms or Trojan horses, and 'Unused, unsolicited Internet traffic', which is considered to be Internet traffic that is received without having been requested and which remains unused by the receiving system (this group may include some worms such as CodeRed – here the traffic is malicious as well as unsolicited and unused). All traffic types are taken into consideration, excluding SMTP (email) traffic.

On a macroscopic scale, any Internet bandwidth that is used for malicious traffic or unsolicited traffic can be regarded as Internet 'pollution'. The firewalls that were used to provide the basic data for this paper are able to distinguish between non-malicious solicited Internet traffic and unused, unsolicited and/or malicious Internet traffic.

Monitoring Malicious Internet Traffic

There are several different methods that can be used to acquire data on Internet traffic. Every method has its advantages and disadvantages and some are better suited for some purposes than others.

Between May 2001 and March 2002, *Symantec* conducted surveys to gain a better insight into the end-user firewall market. Each participant was asked to fill out a questionnaire in addition to submitting their firewall log files for analysis. The firewalls in question were the versions of *Norton Internet Security* and *Symantec Desktop Firewall* that were current at the time of the surveys.

The main advantage of using the firewall logs to obtain data is that all Internet traffic is parsed and presented in a readily digestible fashion without the need for additional software or hardware. The logs in question present the data in a level of detail that suits the purpose, while remaining very cost-efficient. The main disadvantages of this method are that the firewall logs do not give unlimited detail on Internet traffic and they do not cover email Internet traffic.

The firewalls identified backdoor Trojan probes by port number because most backdoor Trojans open a default port. However, many backdoor Trojans can be configured to use a custom port, representing a margin of error. The outcome of this study seems to support the assumption that most backdoor Trojan probes are executed on the default ports.

The surveys were conducted in different regions within Europe. Each region was surveyed for a month. The regions and their corresponding time frames were:

- United Kingdom: May to June 2001.
- Denmark, Finland, Norway, Sweden: September 2001.
- The Netherlands: September to October 2001.
- France: January to January 2002.
- Belgium: February to March 2002.

The data from each country and time frame provides an insight into regional differences as well as changes in malicious Internet traffic over time. Since a different number of participants were surveyed in each region, all the data is presented as percentages.

Unsolicited Incoming HTTP Traffic

In order to view a website, the user's machine normally sends a request to port 80 (HTTP). When it reaches the machine hosting the website in question, the request is fulfilled by the host sending information back to the user's computer, which receives and displays the website. This is depicted in figure 1 (below).

Figure 1: Normal HTTP traffic handling.

The HTTP request is sent to an Internet address at which a web server must be set up appropriately in order to process the request. The only machines that should receive requests for websites are web servers.

In the log files surveyed, incoming HTTP requests were observed that should not occur normally on end users' machines. Such incoming HTTP packets are considered Internet pollution, because they are discarded by any computer that does not function as a web server.

Types of incoming unsolicited HTTP traffic observed were:

- Worms
- Intrusion attempts
- Search engine robots searching for web servers to index.

When the search engine traffic is eliminated from the data, all the malicious HTTP traffic is left. Figure 2 shows the

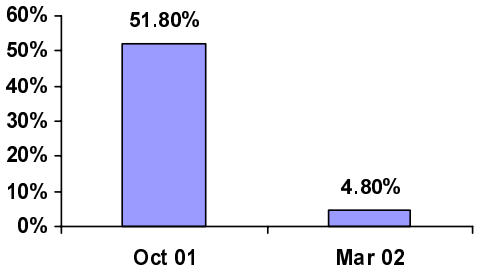


Figure 2: Clear contrast in malicious incoming HTTP traffic.

malicious incoming HTTP traffic for early October 2001 and for early March 2002. The huge amount of malicious HTTP traffic that was observed in early October 2001 consisted of worm attacks such as Nimda and CodeRed as well as hacker robots that attempt to gain access to unpatched web servers.

Less than half a year later, the HTTP worms' contribution was much reduced and hacker robots were the main form of attack. This is a logical development, taking into account the fact that many companies that had vulnerable computers have deployed patches and implemented proper AV strategies.

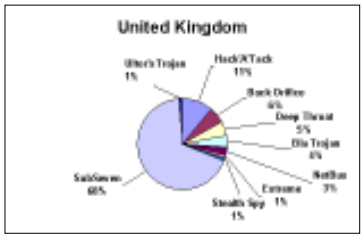
The above indicates that the Internet transported less malicious HTTP traffic in March 2002 than it did in October 2001. However, the amount of irrelevant incoming TCP/IP packages of non-malicious nature has risen tremendously. Most of this data is generated by peer-to-peer networks searching for servers. This means that the Internet's bandwidth is becoming increasingly polluted with data that will be discarded on arrival at their destinations.

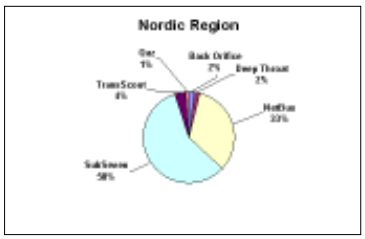
Backdoor Trojan Probes

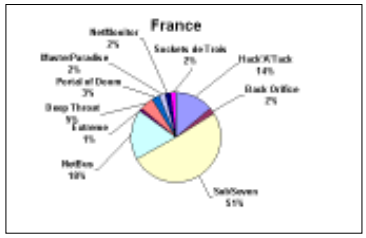
Backdoor Trojan horses are used worldwide by many hackers. Most of these hackers use standard tools that sweep IP ranges to find vulnerable and compromised computers on the Internet. In order for a computer to be compromised and vulnerable to a successful hacker intrusion, the user must have run a backdoor Trojan. These programs are distributed widely and are most commonly acquired through chat networks that allow file transfer. Backdoor Trojans are often presented with file names that conceal their true nature. Once such a program has been run on a computer, that computer is open to hackers and a hacker can exert a certain amount of control over the victim's machine, depending on the Trojan used.

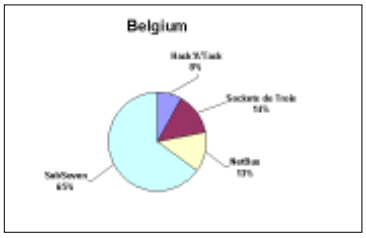
The log files in our research were recorded over different time frames and in different regions. Looking at the IP addresses in the logs, it appears that many intrusion attempts originated from computers that access the Internet from the same Internet Service Provider. This means that the leveraged data is particularly suitable for studying regional variations.

The most obvious similarity between the countries is that SubSeven is the most popular backdoor Trojan among









hackers. Second place is taken by NetBus, which appears to be particularly popular in the Nordic region.

We found the third most common backdoor Trojan was Hack'A'Tack, which is moderately popular everywhere with the exception of the Nordic region.

The remaining percentages are distributed randomly among several backdoor Trojans. However, these numbers are too low to draw a significant conclusion.

Starting from version 2.0, NetBus has become a legitimate commercial product. Although it is not possible to distinguish between different versions of NetBus clients in the firewall logs, we know (from the questionnaires filled out by participants) that none of the users in the study were intentionally using NetBus on their PCs. This means that, regardless of which NetBus version was detected, they were all 'attacks' (or probes).

Conclusion

The composition of backdoor Trojan probes shows that there is an overall tendency for hackers to prefer SubSeven over other backdoor Trojan programs. Despite regional differences in detail, the runner up is NetBus.

The intensity of malicious HTTP traffic seen in the firewall logs has declined significantly since October 2001. This is probably due to the fact that improved safe computing practices have been implemented by many companies since that time. However, while the volume of malicious HTTP traffic appears to have decreased, the overall pollution of TCP/IP traffic has increased since October 2001, with the increasing use of peer-to-peer networks being a major contributing factor.

FEATURE 2

In Search of a Better Way – Performance Testing of AV Scanners

Tony Kwan and Trevor Yann
Computer Associates, Australia

Performance statistics present a number of challenges for both the product evaluator and the reviewer. The person evaluating the product needs to be able to interpret the performance statistics, determining how well they indicate likely performance in the target environment. The reviewer needs to present useful information, without overwhelming the evaluator. The testing and measurements need to be viable for a reviewer to perform, and preferably possible to automate.

Third-party performance reviews cannot pretend to mirror the environment of an end user. Instead, tests are run in their own standardized environment, allowing comparison of performance results within that environment, but not necessarily providing statistics that will indicate which scanner will show the best performance in a particular user environment.

It is not entirely clear how these third-party results should be used. If they have limited applicability, what can be done to improve them? Is it possible that the performance ranking shown in third-party reviews may be different to the performance rankings if the product were tested in the end user's environment?

Quantifying performance is difficult. The performance of AV file scanners is a function of many factors, including hardware, operating system, application use, test set and product features. Performance statistics reported in third-party reviews can be used as an indication of the performance in an end user environment only when the factors contributing to performance are the same between the review tests and the end user's environment.

Hardware and Operating System Differences

Third-party reviews tend to compare AV scanners on a specific operating system, with the same hardware being used to test all of the scanners. It is possible for differences in hardware between the reviewer's test environment and the end user's environment to produce different results. The cost of I/O may be different.

Where the cost of seeking within a file is markedly different, review results may have limited applicability to the end user. An example of this may be an end user that intends to scan CD-ROMs. The seek operation for a CD-ROM is so

slow that it is possible that a scanner that does few reads, but reads many bytes each read, may outperform another scanner that reads a smaller total number of bytes, but has a higher number of reads.

Review results for one operating system may not apply to another operating system, even if the same AV scanner executables are used. The operating system influences the CPU time available to each process, and controls caching of file access, and this can affect the I/O time.

Whilst performance testing is done on a machine that is running few other applications, there are a variety of processes that always run. The implication is that results from a *Windows XP* scanner review may not apply to *Windows 9x* machines, and may also differ from those measured on *Windows NT* machines. Product evaluators need to consider differences in hardware/operating system carefully when comparing performance figures from reviews.

File Usage Patterns

In an end user environment some files may be used many times. This can lead to some or all of the file contents being cached by the operating system. Where different scanners have different mixes of CPU time and I/O, it is possible the cached and non-cached file scanning may give quite different results. Even with a single scanner, the performance may vary between the scanning of files that are cached and those that are not. It is possible that a performance ranking of AV scanners may differ depending upon whether the files are cached or not.

For on-access scanning the performance of an AV scanner might be measured best by the cached scan times, provided the application accessing the file reads a superset of the file parts accessed by the AV scanner. An example of such an application is a file backup. Although the AV scanner will cause physical disk access when scanning the file, the file content is cached by the operating system. When the application subsequently reads the entire file for backup, it will read the parts previously accessed by the scanner from the cache rather than from disk. Hence, the scanning of the file does not contribute to extra physical disk access.

For a scheduled scan of files on a fileserver, the non-cached scan performance is the best indication of the end-user performance. A large number of files will be read, many of which may not be cached.

Test Set

Performance statistics on any set of files are only strictly applicable to that same set of files. The files most frequently used by an end user include a large number of standard operating system and application files. All the AV vendors are aware of these files, and it is not unreasonable to expect performance to be adequate for these. There is more variation to be found in scanning unexpected files.

The set of files being scanned is different under different circumstances. This is particularly evident when comparing on-access scanning with on-demand scanning.

The top ten file types in order of occurrence (most prevalent first) are shown below for three different configurations of *Windows 2000* on a developer's machine. This test looked only at the file extension to determine file type.

Files accessed during booting:	.dll .lnk .sys .exe .ini .ttf .txt .fon .dat .ax
Files accessed during normal usage:	.h .dll .dbx .ini .exe .sys .url .ttf .obj .ntx
Files on machine:	.h .c .dll .obj .exe .lib .html .sbr .txt .gif

Each set of file types is different, although there are some common file types such as .exe and .dll. Scanning performance under each of the different situations relies on the scanning performance for different types of files. In all cases there is a significant amount of access to file types that fall outside the standard exe/ole/script infectable files.

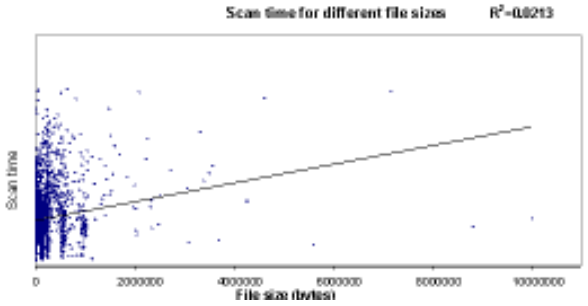
Product Features

Performance can be affected by the scanning features of an AV product. For example, a scanner can cache the results of previous scans. As long as the file is not written to and its data file is not updated, the file need not be scanned for viruses again. This technique to avoid scanning files that are known to be clean can be applied to both on-access and on-demand scanning. A user may find a scanner with such features to have superior on-access scanning performance, even though its on-demand scanning scan speed is slow.

Reporting Performance

How should performance be reported? Time-based units such as files per second can be used. Alternatively, throughput-based figures such as MB per second can be used. Average figures for scanning many files can be used, or figures for each file scanned.

If the throughput is calculated by dividing the total size of the files scanned by the elapsed scanning time, then there is an assumption that the scanning time has a linear relationship with the file size. We tested this assumption by plotting the scan time per file against the file size, and trying to find a linear relationship between them. The results (see figure



below) are from a normal scan of uncached files (after a system reboot). A line through the plot illustrates the best linear relationship that could be determined. The R^2 value is a measure of how good the linear relationship is, with 1.0 indicating a perfect relationship. The R^2 value shown (0.0213) indicates that, in this case, there is only a very weak relationship between the scan time and file size (hence, in the plot shown the line of 'best fit' is for illustration purposes only).

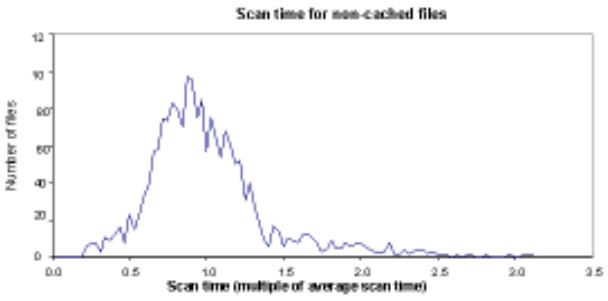
This is not entirely unexpected. Most AV file scanners will not read the entire file to determine whether or not it is infected. This is particularly evident for many OLE files, where the lack of macros can be detected easily and quickly, without regard to the size of the file.

It is possible that when the scanners are operating in full/reviewer/paranoid modes they may read and process most of the file, and therefore show a stronger relationship between file size and scanning time.

Where it has been established that there is a linear relationship between file size and scanning time, throughput is a valid measure of the performance. Where file size and scanning time are largely independent, then performance would be better reported using time-based figures.

Published performance figures are usually average figures calculated from scanning many files. As an alternative, per-file timings allow the distribution of scanning time to be determined, and the identification of those files that are scanned slowly. On-access scanning on an end user's machine may be affected by a long scan time on a frequently-used file.

The average scan time does not indicate whether there are many files that have extremely long scan times. The plot below shows the distribution of scan times around the average scan time.



The distribution shown is largely symmetrical, but has a tail showing a number of files whose scan times are more than 1.5 times the average scan time. If any of these files are accessed with an on-access scanner enabled, the performance hit may be quite noticeable.

Measuring scan time on a per-file basis is difficult. Excluding scanner load/initialization times, we found there was a wide variation in scan times, necessitating a large number of scans in order to obtain statistically valid scan times.

The solution used to generate the results shown here was to have the scanner callable as a dynamically linked library. The test program separated the initialization and scanning operations.

A Better Way?

The task of measuring and publishing performance results is difficult. At the very least the test environment should be described in detail, to allow product evaluators to consider the applicability of any statistics produced. This should include a description of the variation in the files that are used in the test set.

Based on our investigations we recommend that:

- Both on-access and on-demand scanning performance should be tested.
- Multiple test sets should be used, including files that are not traditionally infectable (such as GIF and PNG). One or more test set(s) should be composed of files that are part of operating system (or popular application) installations.
- Throughput statistics should only be used after it has been established that there is a strong relationship between scan time and file size. Where there is not a strong relationship, time-based statistics should be used.
- For on-access performance testing the selection of the test set should include files that are accessed during booting and during normal usage.
- On-access scanning performance should include an indication of the performance measured for those files that are scanned most slowly.
- Results should be measured and reported for both cached and non-cached files.

If all of these recommendations are followed, a large set of performance statistics will be available for publication. The usefulness of the statistics needs to be balanced against the difficulties in obtaining the measurements, and the danger of overwhelming those reading the reviews with too much information.

A valid interpretation of the performance figures is centred on how well the test environment matches the intended production environment. Only after careful consideration of this match/mismatch of environments can the performance results be interpreted appropriately, enabling the identification of AV scanners that meet the performance requirements of the user.

Technical Details

All test data in this article were obtained from a pre-release version of a commercial scanner with additional diagnostic output, running on Windows 2000 SP2. *The test machine was PIII 1G with 256 MB RAM.*

OPINION 1

Knight in FUD Armour

Pete Sergeant

Microsoft began releasing information about its *Palladium* system at the end of June this year. Most of the information *Microsoft* has revealed has been channelled through its marketing and PR departments – solid facts are few and far between, impressive claims are abundant. In the first official article about the project, words like 'apocalyptic' and 'jihad' were tossed around with reckless abandon.

Palladium promises a lot of things, which are very carefully worded by *Microsoft* PR. Allegedly, *Palladium* will: stop all viruses and worms, stop spam, safeguard your privacy, protect your personal details, and protect your intellectual property.

Breathless pieces of spin talk about how media companies can make sure users don't steal their data, and how you can be sure that your employees won't leak your sensitive data. But, if you're anything like me, one item will have caught your eye: *Palladium* will be the end of viruses and worms on systems that run it.

Palladium uses a mixture of software and hardware control (the hardware element based roughly on the Trusted Computing Platform Alliance specification) to ensure that if you've booted your computer in trustworthy mode then you will only be able to run signed software components. The logic follows that, if you're a virus or worm writer then getting *Microsoft* to sign your code will be a somewhat difficult process.

Additionally, and details are scarce in this area, different parts of the system will be protected from each other – that is, your copy of *MS Word*, when exploited, won't be able to similarly taint and disrupt other components on your system, nor will it be able to steal your credit card and *MS Passport* information, because ... well, it just won't. Honest.

A Dose of Reality

This all sounds awe-inspiring. If the system works as intended, we will never have to worry about computer security again (although there will be some 'irksome side-effects', see below). Anti-virus companies can cease operations or start concentrating on the less secure operating systems (Mac OS X, *Linux*, FreeBSD), and the world will be a happier place. You'll even have to put up with fewer stories about the world being brought to a halt by 'cyber geniuses' and their 'incredible teenage hacking skills', and the accompanying Rosenberger counter-rants …

Personally, in Bill I do *not* trust. *Microsoft* has a very poor history of writing good quality and bug-free software. The assumption is made that a virus will not be able to execute on the system, because it will not be signed, and that this safeguard is insurmountable. Of course, having not been told many specifics of the system, this point is hard to either back up or dispute: it will have to simply suffice to say 'wait and see'.

Let's assume this does work – foreign executables don't run. But what about worms that access the system through software that *is* trusted (*Outlook*, *IIS*, *SQL Server*)? How will *Palladium* protect against this? Apparently the plan is to sandbox applications in, so, even if the applications are exploited, the malicious code cannot affect other parts of the system. How reassured I am by knowing that only my data will be able to be modified/deleted/propagated by worms, rather than other parts of the system is left for the reader to guess. There seem to be no safeguards in the system to protect against worms.

An interesting side-effect here is how anti-virus software will interact with other parts of the system – presumably it too will be sandboxed in to protect it from modifying data it hasn't been told it can have access to, like, say, a confidential *MS Word* document (protected, of course, by *Palladium* digital rights management) that's infected. At the very least, it's going to force all the anti-virus vendors to 'jump into bed' with *Microsoft* to ensure their software will be allowed to run with sufficient privileges on *Palladium* systems, quite possibly entailing many heavily disguised channels for cash extortion along the way.

Whether you're for or against the 'virus protection' offered by *Palladium*, some of the 'features' it will offer have rather nasty side effects, for example the total control of your computer by *Microsoft*.

Assuming *Intel* and *AMD* get in on the act (as they are already doing), and start producing '*Palladium*-enabled' systems, you stop really having a choice about what software gets run on your machine – it's either blessed by *Microsoft* (think: corporate buddies) or deemed to be 'untrustworthy' (think: open-source, GPL). While, in theory, anyone will be able to submit code for *Microsoft* to evaluate and sign, the cost is likely to be prohibitive. And if you decide to modify your open-source software, it'll no longer be signed, and your computer may refuse to run it.

Conclusion

A lack of many hard details about the system prevents any real probing analysis of it. However, it would seem, to the cynics amongst us, that *Microsoft* intend to use the security theme, mixed with a healthy dose of user Fear, Uncertainty and Doubt, to inflict a system upon the world that would give it micro-control of their computers.

OPINION 2

Linux Malware: Debunking the Myths

Phil d'Espace

As Eric Raymond detailed in *The Cathedral and the Bazaar*, open-source software is written by programmers who are motivated more by a sense of community than by financial gains. As a result, the open-source community is a strong one, yet one that's accessible to almost anyone with a computer and some free time. This community holds an almost magical appeal for the disaffected youth who desperately want to belong – in effect, there's a tendency for many to become like the sheep in Orwell's *Animal Farm*; loud and vocal supporters of the free system, without any real understanding.

The resulting ethos of repeating misunderstood 'advocacy points' has led to the perpetuation of a number of myths. It is not surprising, given that *Linux* – the flagship open-source operating system – lacks many of the niceties associated with a modern desktop operating system, that many of these myths are based on *Linux*'s fabled high level of security.

A common belief appears to be that *Linux* is somehow impenetrable to viruses and worms, as was demonstrated by a couple of zealously written, yet misinformed, letters to *VB* (see *VB*, August 2002, p.4) concerning Peter Morley's article (see *VB*, May 2002, p.16) which touched, amongst other things, on *Linux* malware.

Thus, my mission for this article is to debunk some of the myths highlighted by those letters (and from other sources) concerning viruses and worms on *Linux*.

Myth: '*Linux* malware needs root privileges to spread/cause damage'

The super-user/administrator account on *Linux* is called root. A person or application logged in or running as root has access and ability to modify, delete and create files at will. When you're logged in as root, the operating system has no defence against you – if you want to delete and modify system files, it's your choice. Most system files can be modified only by the root account, hence the idea that, in order to cause damage, a worm or virus needs to be running as root.

However, this largely ignores the great amount of damage that can be done by a program or person running or logged in as an unprivileged user. Not all normal user accounts are equal, and the most vulnerable class are often those that will cause a major headache if data belonging to them is deleted/corrupted.

Consider the *Apache* web server – another of the flagship products of the open-source movement. Fairly recently, *Gobbles Security* found a security hole that allowed an outsider to gain access to the machine in the guise of the user that the web server runs under. On most websites, this user account will have read-access to all the files on the website, and quite possibly write-access to others. If a worm enters through this point, and drops a JavaScript/VBScript/ActiveX/whatever worm or virus into your HTML files, or even simply replaces them with some 'u h4v3 b33n Hax0red' message, you will encounter some serious problems, and I'd certainly count this as damage.

What about mass-mailing worms? Can we assume that no *Linux* mail clients will have buffer-overflow vulnerabilities?

As the number of *Linux* users increases, as does the number of users of *Outlook* clones like *Ximian*'s *Evolution* and *Mozilla Mail*, this seems a very real possibility – Mutt (a popular text-mode Unix mail program) has had several buffer-overflow bugs that (in theory) allowed the execution of arbitrary code, and that has the luxury of not having to worry about a complicated GUI and HTML rendering. No root account is needed to exploit a bug like this, so data destruction is possible, etc.

That's email worms covered – what about malware that spreads in a different way? An *Apache* worm was released shortly after the *Apache* vulnerability was made public, and the Morris worm exploited a vulnerability in sendmail, causing a huge amount of damage.

So, while it may be difficult for *Linux* malware to infect files to which it doesn't have write-access, this neither stops it from causing damage, nor from spreading to other machines – nor even from lying dormant until, for whatever reason, the root user executes it.

It's worth mentioning at this point that, once you get into a machine as an unprivileged user, privilege escalation can be relatively simple – many vulnerabilities exist to allow this and more crop up frequently. A Hybris-esque virus could just camp out on the machine and wait for plug-ins to exploit the latest local exploits. Then, as root, anything is possible.

Myth: 'The root account won't run malicious code'

Of course, the first myth forms from another – that it's hard to get root to run infected code. As Curtis H said (see *VB*, August 2002, p.4): 'all of these defences go out of the window when [you use] root as [your] main user account'. This assumes that root doesn't do something stupid (nor is led to do something stupid) during routine system maintenance. There are a number of situations in which this can be true, but I shall pick a common one as an example.

It comes down to laziness on the part of system administrators. Of the packages a home user, business user or any user downloads from the Internet to install on their machine, how many do you think are checked for malicious code? Answer: virtually none.

First, not many people have the skill – or the will – to check through thousands of lines of code to find a tiny piece of embedded malware, or perhaps an ELF infector. Secondly, many people assume that *other* people are checking through the code, and that they don't need to do so themselves. A fair assumption? Try asking anyone who installed a trojanized copy of irssi or open-ssh as root.

As a brief aside, Lindows, a *Linux* distribution aimed at non-technical users, uses root for the default account. One would hope the use of Lindows doesn't become prolific.

Myth: 'Open-source software is inherently more 'secure', and so there are fewer exploitable bugs for malware to take advantage of …'

While strictly true, this is very misleading. That it is easy to find bugs in open-source software is both a blessing and a curse – while more bugs are fixed, it is easier for black hats to find the exploits in the first place. Security through obscurity may be knocked by open-source advocates, but it does keep some vulnerabilites permanently hidden.

Open source advocates are very keen to shine light on the quick patching of open-source projects. Again, this is normally true.

However, the mere fact that a patch has been released doesn't mean that everyone running the piece of software will hurry to upgrade. People may not even know they have the software running on their machine, or they may not like the idea of downloading and recompiling the software – *Netcraft* claims there are still very many exploitable *Apache* servers on the net.

Finally, and as mentioned previously, you can check the source code of anything you install on your machine for viruses or other malware. It's a nice thought, it really is, but very few people have the time, inclination or skills necessary to do this – instead, there is a warm glow of false security.

Myth: '*Linux* forces you to be more security-conscious, and thus you won't be gullible enough to get a virus if you run *Linux* …'

Linux forces you to be more security-conscious? Mmm, especially those distributions that tout themselves as being simpler to install than *Windows*, I'll bet! When *Linux* was still difficult to install on your machine (I remember finding it pretty easy when I first started using *Linux* in 1997), the extra work needed to install and run *Linux* tended to keep those who weren't prepared to tweak their system excessively out of the loop.

Nowadays, the mouse hand clicks the pulsating install button while the other wipes drool from the mouth, the brain having been faced with the choice between 'server' and 'workstation'. Trust these people to keep their systems up-to-date to a sufficient degree not to get viruses? No, me neither. What's more, the rabid advocates have been telling them that their machine is extra-secure from day one, so they get another lovely shot of the false-security drug.

Myth: 'ELF isn't an infectable file format'

Granted, this one isn't heard very much, but some people do believe it. Reasons given are normally specious arguments about entry-points and other examples of a little knowledge going a long way, while all the time completely ignoring the existence of current and very real ELF infectors. Essentially, this holds a light to the candle that burns with the message 'people will believe what they're told'. When they're told that *Linux* is uninfectable, and are so desperate to believe it, they'll repeat anything.

In Conclusion

The sole reason we don't have a *Linux* malware problem is lack of a sizeable user base. Yes, there are some protection mechanisms in place; no, they aren't insurmountable. To say blindly '*Linux* is safe from viruses and worms' without examining the facts first is to boldly go where too many mindless Slashdot readers have gone before.

VIRUS BULLETIN **©2002 Virus Bulletin Ltd, The Pentagon, Abingdon, Oxfordshire, OX14 3YP, England. Tel +44 1235 555139. /2002/$0.00+2.50**

PRODUCT REVIEW

GFI MailSecurity for Exchange 2000 VS API Mode

Matt Ham

GFI produces a wide range of communication-related applications, of which *GFI MailSecurity* is one of a number of security-related programs.

Other products in *GFI*'s range include: *GFI FAXmaker*, allowing fax use though *Exchange*; *GFI MailEssentials*, offering disclaimers, archiving, content filtering and the like; *GFI DownloadSecurity*, which filters FTP/HTTP on *ISA Server*; *GFI LANGuard Security Event Log Monitor*, an IDS system; and *GFI LANGuard Network Security Scanner*, a security hole scanner.

With such a wide coverage of security issues it came as no surprise that the *MailSecurity* program contained features which could be included easily in some of the other products' spheres of influence.

GFI MailSecurity

GFI MailSecurity is described on the company's website (http://www.gfi.com/) as 'Email content checking, exploit detection and anti-virus.' Three years ago there would have been a fairly clean division between these three features – nowadays, however, the three are all facets of the same problem.

Admittedly, of the more usual desktop scanning engines reviewed in *VB*, only two of a field of 20 or more have made distinct and memorable mention of exploit checking. However, it is anticipated that this number will expand in the coming year, and the exploit checking seen here might be a good forewarning of what is to come.

Although a subject of hot debate at last years's *VB* conference, content checking is now a mainstay in first-line anti-virus defence in many companies. Therefore, the area covered by this review strays away from the more traditional provinces of a standalone review, yet is as relevant as ever.

VS API versus SMTP

GFI provides one program which will install on either an *Exchange 2000* server in VS API mode, or as an SMTP message scanner. If installed in both manners there should be no interoperability problems and different settings may be applied to each installation. Where the advantage lies is very much dependent upon the network architecture of the implementing organisation. In this case, the comparison will be made of two implementations, both using *Exchange*, but one exercising filtering through SMTP while the other uses VS API.

From a practical point of view, the VS API solution is by far the more plug-and-play of the two. VS API requires no configuration during installation other than supplying an administrator mail address.

The SMTP-based product, on the other hand, requires the setting up of a separate physical machine for the *MailSecurity* program as well as the alteration of a considerable number of parameters in the SMTP configuration.

As a pointer to the difference in the amount of configuration required, the section on installing the VS API version runs to three pages in the manual, which includes one page of system requirements, while the SMTP installation section takes up a rather more lengthy 12 pages.

As far as performance is concerned, there can be expected to be some speed advantages to using an inbuilt API within the *Exchange* system as opposed to an external filter which requires additional routing of messages through a physically separate server. However, these overheads due to physical separation are potentially not as cumbersome as might be expected. The *MailSecurity for SMTP* application, although needing to be loaded on a different machine, can share this with other applications – for example a firewall, which would also usually be physically separated.

The most likely features which will be taken into account when deciding between the two options is the matter of what can be scanned and the limitations of VS API when applied to externally routed messages.

Since it is located within *Exchange*, the VS API scanning method is by far preferable where internal message scanning is required. Although, theoretically, it would be possible to re-route all internal mail through an SMTP server, in practice this would be a waste of resources and thus the VS API version is all but inevitable.

The VS API version does, however, have one important flaw where external mail is concerned. Due to limitations in VS API it is necessary to forward manually all mails which have been initially detected as breaking one or another rule, since the exact external end destination is not preserved in all cases.

Thus the best option would seem to be to implement both systems, with the SMTP version scanning inbound and outbound mail from the organisation, while the VS API version scans internal mail. This also allows for the implementation of different sets of rules for internal mail from those imposed upon mail entering or leaving the organisation.

Installation

The system specifications given for the product are quite narrow, being the standard or advanced version of *Windows 2000 Server* with *Exchange Server 2000* installed, both with service pack 1 or higher. The insistence on *Exchange 2000* is not particularly surprising, since this was the first *Exchange* product to include VS API.

For the purpose of testing, *Windows 2000 SP2* and *Exchange Server 2000 SP2* were used. Mail clients used were the *Windows 2000 Server* and *Windows XP*-installed version of *Outlook 98* and the *Windows XP Professional* pre-installed version of *Outlook Express*.

GFI MailSecurity was downloaded from *GFI*'s ftp site as a single zip file of around 11 MB, which decompressed to a single EXE file of much the same size. This can be used to install on either SMTP or *Exchange*, the latter being triggered if the machine has *Exchange* installed upon it.

The similarity in size of the files is explained by this being a compressed executable, as announced when it is executed, stating that the EXE drops an MSI file for automatic execution. *GFI* is fairly unusual at this point in that WinAce – a program rarely used outside the warez community – is used as a compression tool. Despite the talk of MSI files, when execution is complete a dialog appears, asking whether MailSec.exe should be launched, though Windows Installer is launched later.

At this point the installation routine proper commences, with a standard preamble concerning copyright and the meaning of Next and Cancel … thrilling stuff.

The first option available in the installation process is to check whether there is a newer version of *MailSecurity* available on the *GFI* website. This was not selected for the review process, since *VB*'s network of test machines is isolated from the outside world. In the case where new version checking is selected, this can only be performed directly from the *GFI* site, no configuration being available here, and thus is unable to be redirected to a locally updated source.

Next in the installation process is the licence agreement, followed by a registration process in which the serial number is an epic of its type, consisting of 41 assorted characters – quite enough to give me a headache when inputting it. After this the administrator mail account is requested, with a default provided, and the install path is chosen. File transfer then commences, after which installation is complete.

Documentation and Web Resources

Documentation is quite extensive for *GFI* products, most of it being available from the website. The manual for *MailSecurity for Exchange/SMTP* is close to 80 pages long, with the individual sections being page-numbered separately, rather than the manual as a whole. As a reference, the contents pages are rather more useful than the index (and, in fact, have more entries).

The first part of the manual to be used was the section on installation, which proved a very good reference on system requirements and what to expect during installation. The level of documentation cuts a good path between over-patronising and minimalist and, unlike many products, the manual was used frequently as a reference to functional aspects of the product during the review process.

There were additional sections in the manual covering the pros and cons of installing the product as the VS API version as opposed to the SMTP version, as well as the method of installation for the SMTP version.

GFI's website contrasts with that of most dedicated anti-virus vendors, which is explained by the fact that the strictly anti-virus part of *MailSecurity* is provided in conjunction with other companies. Instead of hosting the usual 'newest alert' or disturbingly common 'spurious press release designed to get publicity through a hysterical doomsday scenario', most of *GFI*'s home page is taken up with links to product and marketing information of a more down-to-earth nature. Much of this is presented in the form of white papers, traditionally the preserve of companies who wish to market without being seen to be too pushy.

Somewhat hidden within this businesslike site are two areas which match up to more traditional expectations of an anti-virus site. First is the 'Knowledge Base' and second the 'GFISecurityLabs'.

The Knowledge Base turned out to be most akin to a search engine dedicated to the various white papers, documentation and troubleshooting information for *GFI*'s range of products. This is up to date judging by the fact that it includes entries for products released only recently by *GFI*. Its usefulness was, thankfully, not tested first-hand since the *MailSecurity* product was well behaved. However, an inspection of the information gave the feeling that the troubleshooting information was quite extensive. Using the crude method of searching for 'virus', for example, produced 20 pages of references.

In contrast, GFISecurityLabs are less up to date. The majority of the security alerts listed on the page are virus-related and run from the relatively recent W32/Frethem.K to the historic W32/MTX, with only six intervening issues being considered worthy of being reported. In an area where new security flaws erupt weekly if not daily, this is a fairly paltry attempt at defining what is a current threat.

The Interface

Upon installation, six objects are added to the *GFI MailSecurity* section of the Start-Progams menu. These are GFI Monitor, MailSecurity Configuration, MailSecurity Help, MailSecurity Troubleshooter, Moderator Client and Register MailSecurity.

GFI Monitor

GFI Monitor is very much a one-trick pony, in that it simply displays the results of scanning. Thus it supplies information on the number of items processed, quarantined items and blocked viruses. Items here are not synonymous with either mails or attachments and are discussed further a little later in the review.

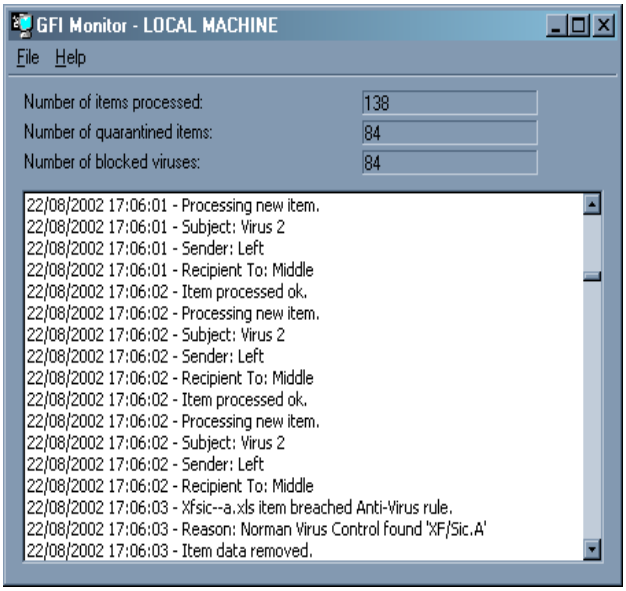
A more detailed summary of what has happened to each item is supplied in a scrolling window. This offers details of email subject, sender and recipient, and the results of the *MailSecurity* scan. The results can be such events as, for example, the item breaching a rule, the item quarantined, or the item data removed.

Although the display does offer useful information, the charge may be levelled that the lack of delineation between reports for different items or different emails is singularly poor. This leads to visual inspection of the Monitor output being a little more difficult than necessary.

MailSecurity Configuration

MailSecurity Configuration is altogether a more complicated beast. It is activated as a Microsoft Management Console (MMC) component, and follows the hierarchical tree structure which that framework encourages. The subsections here are General, Content Checking, Virus Scanning Engines, E-mail Exploit Engine, E-mail Threat Engine, Quarantine Options and Logging. With so many sets of options available not all are absolutely notable and thus such matters as the ability to check version number are not described fully in the review.

Under the 'General' heading, the most interesting feature is the ability to set the way in which VS API is implemented. The option used, termed here 'Pro-Active Scanning', means that all mail objects are placed into a queue as they arrive. Objects are then scanned in that order, unless an attempt is made to access an object in the queue, at which point it will be scanned immediately before access is allowed.



An alternative implementation is 'On-Demand Scanning', in which messages are scanned only when a request for access is issued. In addition there is also a setting which will scan all objects in the *Exchange* message store. This option is termed 'BackGround Scanning' and is documented to cause significant server load if implemented on a large *Exchange* store. A manual scan is also available here, which, again, runs through the entire *Exchange* store.

The 'Content Checking' heading opens a tabbed dialog. Although not a dedicated anti-virus function, content checking can be used for front-line prevention of many new email worms, often before anti-virus companies have released thoroughly tested virus descriptions. In addition, hoaxes such as jdbmgr.exe can be filtered in a crude but effective fashion. For these and similar reasons the content scanning portion of the software is worthy of comment.

Content checking for body and attachments can be specified either by a keyword or condition, conditions being simply keywords linked by boolean operations. It is not stated, however, where precedence lies and parentheses are not supported. Since A OR (B AND C) OR D is not at all equivalent to (A OR B) AND (C OR D), a little more clarity here would be helpful. However, it is possible to set precedence on the order in which rules are applied.

In any case it will be pointless to check a PGP-encrypted mail in this manner and thus it is possible to block all such mail at this point, though the default is to allow PGPed mail to pass through. It is also possible to set checking to apply to combinations of inbound, outbound and internal mails, which can each be selected independently.

Subject lines are not as finely tuned – no boolean operators can be selected. Like the body and attachment filtering, however, rules manufactured here may be exported and imported, facilitating a standardisation on certain filters throughout an organisation.

Also applicable to both these features is the reaction to trigger when a banned word or word combination is detected. These are somewhat akin to those available on a standard desktop scanner for the treatment of infected objects, though with one noticeable omission. Notification of the user or a designated manager is possible, as is the logging to file of exactly which rule was infringed. In addition, the email breaching the rule may be blocked and quarantined or deleted. However there is no option for disinfection, since this is hardly applicable to what are assumed by their subject or contents to be wholly worthless. As such, the situation is strictly more analogous to the usual treatment of a worm than to a virus.

Last but not least in the Content Checking section is the ability to determine exactly where these rules will be applied. Rules may be applied either to specific mail folders and users or to all but those specified. This does leave the

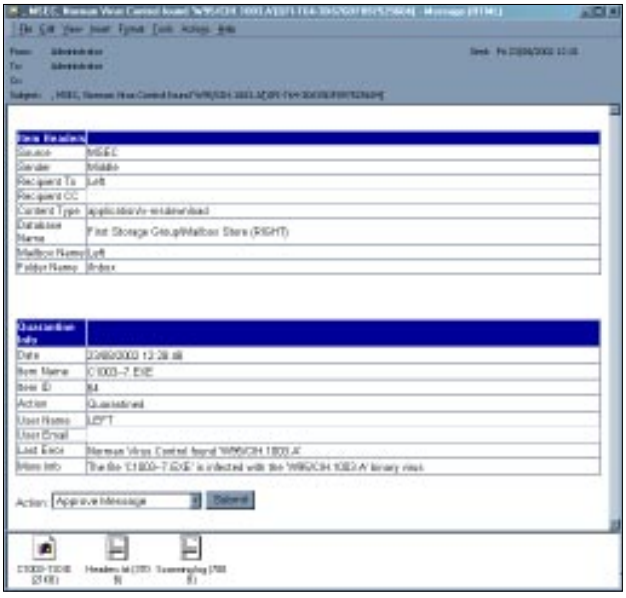
degree to which control may be exercised as very much on or off for a specified location, since although multiple rules are available, they cannot be linked to specific user sets or sets of excluded users.

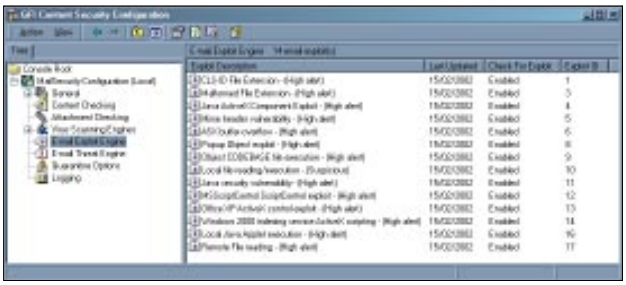
Of a similar nature is the Default Attachment Checking Rule Properties dialog. Again, this may be applied to inbound, outbound and internal mails in any combination, though the default is to check only inbound attachments. While perhaps there are reasons for not excluding executable file attachment transfer within an organisation, it would seem to be tempting fate to embark upon a policy which allowed free transfer of executables from within to contacts such as customers.

The default mode of operations is to block a selected list of attachments which fall under the wide category of executable, ranging from the aged COM file to the more modern MSI. Somewhat singled out among *Office*-related extensions, MDB files are set as blocked by default, though the same is not true of DOC, XLS and PPT files. The default situation also makes no mention of compressed file types other than CHM.

As an alternative, a list of permitted file types may be used, or all files may be banned as attachments. The areas for actions to take upon attachments which break rules and those defining which users are to be subject to these rules, is identical with those for content checking. Again, these may be exported and imported.

Into more familiar territory, the engine configuration for those supported is next in the interface. Considering the differences inherent between the *BitDefender*, *Norman* and *McAfee* engines, it is surprising at first glance that each has an almost identical control interface, and indeed that these interfaces are so simple. Presumably this is in the interests of streamlining the process of integrating the different engines within the *GFI* interface, and possibly explains





the default blanket banning of MDB objects as noted previously.

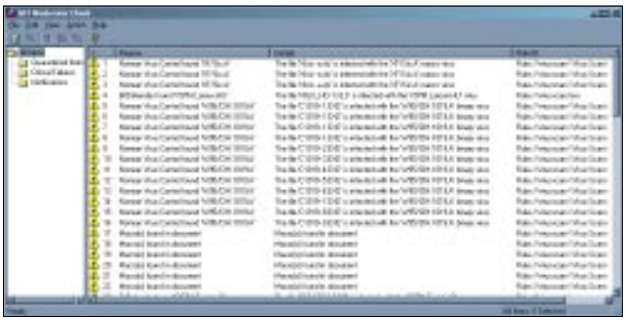
The default settings for each engine are to scan all mails, be they inbound, outbound or internal. They are all also set to block all documents containing macros, the treatment of macros being the area in which the engines differ. *McAfee* and *BitDefender* are both noted to block all *Word* documents containing macros in this configuration, while *Norman* extends this to *Word*, *Excel* and *Powerpoint*. Since *Access* is not mentioned on any of these configurations it seems likely to be unchecked for dangerous internal code and thus worthy of rejecting in its totality.

The Virus Scanning Engine configuration is limited to this much. The related matter of which download sites should be used for updates is also controlled in this area. Since *GFI* uses API interfaces to the scanning engine rather than the usual consumer products, the source of these updates is not the engine developer websites but the *GFI* ftp site.

Continuing with the Content Security Configuration interface the next topic is the E-mail Exploit Engine. This is a hard-wired set of exploits which the scanner will attempt to detect in any mail passing through *MailSecurity*. Each exploit detection is given a priority, date of update and can be selected as on or off independently of the others listed. Currently the list of exploits numbers 17.

As an example of the exploits covered, exploit ten is noted as covering 'Local file reading/execution' and this is flagged as suspicious. What is missing here is a more detailed explanation as to quite what each exploit is. While willing to accept that local file reading and execution is potentially a very bad thing, what will I do when this alert is triggered? As a responsible administrator I will, of course, be concerned as to whether I have already made this particular exploit a moot point by use of available patches. Without more exact definitions of each exploit an administrator is left in the irritating position of knowing that an exploit is being attempted, but not specifically which one of several that might be described in vaguely the same terms.

Approaching the final strait we reach the E-mail Threat Engine Properties, a dialog which is far from self-explanatory, since context-sensitive help is unavailable throughout the Content Security Configuration interface. Reference to the manual reveals that this is a feature which is currently in development, with the HTML script defuser being the first



portion to reach fruition. As it stands the defuser scans HTML files for scripts and removes these, assuming that, if not malicious, they are at least expendable. This process can be tuned from the default of removing all HTML scripts, so that only 'highly dangerous' objects are removed. Unfortunately, like the exploits mentioned before, quite what defines 'highly dangerous' is not mentioned. By default this is applied to internal, incoming and outgoing mail.

The Quarantine Options properties dialog, on the other hand, is self-explanatory. When a mail is designated as needing to be quarantined the onus lies upon some human to decide what should be the ultimate fact. Each user can be assigned a manager, thus allowing delegation of this task, or a central administrator can be appointed for all incoming mails. Clearly the latter would be reasonable only in smaller organisations. Alternatively, quarantined objects can be sent to a public mail folder – for inspection and processing by a team dedicated to a user base rather than specifically assigned users.

Finally, the logging section is covered by four files. The targets for log output may be altered as desired, the contents covering: anti-virus, attachment checking rules, content checking rules and email exploit detection logging. While the on-screen monitor was criticised for multi-line reporting, the log files assign all the information on one line for each incident. However, such matters as user and source are not logged and thus detailed analysis of the statistics held within the log files will be hampered.

MailSecurity Help and MailSecurity TroubleShooter

It was noted in the previous section that *MailSecurity* is distinctly lacking in a context-sensitive help function, relying instead upon this separate help application. This is all well and good, but in practice the help application is no more than a searchable version of the manual. The addition of a search feature does enhance the usability of the manual, but this is no real substitute for a context-sensitive source of information.

Similarly, the troubleshooter delivers less than its name might suggest. Rather than offering the *Windows* native variety of interactive, if usually useless, troubleshooting information, it is more of a simple form allowing problems to be stated, system information gathered and the whole of this information gathered into one handy zip file. This file is then to be submitted to *GFI*'s technical support department. Although a useful method for gaining support, this is not really what would be expected from a trouble-shooting wizard.

GFI Moderator Client

Back once more into the realm of functionality, this application is designed for the administrator to control the fate of those messages which are sent to the quarantine. The messages are displayed in list form and may be selected individually or grouped by control clicking. Objects are in fact not limited to quarantined items but also include critical failures and other notifications, though in the course of review no critical failures were noted.

Considering quarantined items, each has an item ID, reason for quarantine, details, date, user, user email, type and rule ID associated with it. Selecting a line from this list gives the full email of which the item was a part.

When the object has been selected it can be marked down for approval to be forwarded to the original target, deletion and notification of the same, or simple deletion. Since the list of objects can be sorted by reason for quarantine and bulk object selections made for treatment, it is possible to administrate many emails at the same time. This also allows for more detailed examination of those mails which are of a more interesting nature.

In Practice

The first test used was a simple case of sending through an email with no attachment and no unpleasant content of any sort. Thankfully this was transmitted with no problems and showed up as having been a scanned object on the *MailSecurity* monitor.

This was followed by sending a known clean exe file through the server – again with the result that the mail reached its destination. In this case, however, three objects were noted in the monitor as having been scanned. The mail itself and the attachment being considered as separate by the scanning process is intuitive, though the third scanned object is a mystery. This is even at odds with the manual's description of the expected activity, making it more mysterious still.

Further investigation revealed that this was not a matter of objects being scanned when they are checked out of the *Exchange* data store, since all the objects are noted simply by sending an object to *Exchange*, and none when the objects are retrieved.

When these preliminary uninfected experiments were out of the way, a virus sample was selected at random from the *VB* stocks, this being XF/Sic.F. When sent, a mail with an attachment duly arrived in the intended mailbox. The attachment was not the viral file but a description of the results of scanning the file, indicating that these showed it to be infected.

In this case the *Norman* engine was that detecting, which begged the question as to whether the same file would be detected automatically by the same engine on every occasion that it was sent. Three sendings of XF/Sic.F later, and the *Norman* engine was the only one which showed in the monitor's listing of detections.

Of course, it may have been the case that only *Norman* could detect these files and so an older and more generally detectable virus, W95/CIH.1019.A was chosen to be sent repeatedly from one client to another. Again, the *Norman* engine was the one which was triggered. It was only when XL/Laroux.DO was tested in the same manner that the *BitDefender* engine was finally persuaded to show its presence.

After further investigation the situation appeared to be that the engine or rule triggered was essentially random, though very much skewed towards *Norman*'s engine where file viruses were concerned. For macro viruses the skew was more towards generic macro detection, with *BitDefender* coming in a close second.

When the detection of macros within a file was sufficient to deny transmission of the file, the same quarantine action was taken as when a virus was detected. In these cases the count for blocked viruses increased, though there was no proof that these were in fact viral.

During the initial tests the count for quarantined items matched that for blocked viruses exactly. Although the testing of *MailSecurity* was not intended as a parallel to the comparative tests, were the clean macro test set to be passed through *MailSecurity*, the result would be 100 per cent false positives given this behaviour.

It was decided to test the effects of multiple infections in one mail. Initially this was performed with two attachments in the same mail – a situation in which no problems were engendered. This was not surprising, since the breaking up of the message into component objects should lead the engine to treat such a mail in the same way as a string of emails.

Given that *MailSecurity* is designed for large corporate environments, the test systems used could not hope to offer an adequate stress test on this front. The best that could be hoped for in terms of posing potential problems for *MailSecurity* was to include a vast number of attachments in one mail, working on the possibility that some internal counter could be caused to exceed or reach its maximum.

To this end, a mail was prepared containing 702 infected attachments, all infected with macro viruses. The first surprise was that *Outlook* itself did not crash when presented with such an offering.

The processing of the resultant mail was not a speedy matter at all, taking ten minutes from start to finish. During this time the Client Machine declared that the server response was '-ERR Unrecognised internal error'. Despite this unpromising start, once the scanning process had been completed the message was duly transferred with a completely stripped set of attachments, each replaced by a notification text.

Archived and stored files were also tested, in this case using the EICAR test file packaged in various different ways. The files used were archived so as to give CAB, XXE, SHS, ZIP, UUE, TAR, RAR, MME, LZH, GZ and ARJ files. All of these were detected as the EICAR file.

More notably, this was the first occasion upon which the *McAfee* engine was seen to be making a detection. The *Norman* engine scored some points by detecting the EICAR_Test_file_Not_a_Virus in these files, while *McAfee* offered the less informative 'Eicar test file' as a detection note.

All the previous tests were performed using viruses in the form of attachments – not a particularly accurate representation of such viruses as JS/Kak. Therefore, a machine was deliberately infected with Kak and used to send messages. As expected, this triggered virus detection, and the body of the message was removed and replaced with a message explaining this.

Conclusion

Within the limits of the tests performed, *MailSecurity* performed perfectly, although tests were not performed on the whole range of features. The lack of an ability to disinfect executables is notable, but unlikely to be a concern except in a small number of cases where disinfection can be performed manually. Scripts and macro viruses can be removed so that disinfection is achieved, albeit not through the use of the anti-virus products. The documentation was of good quality and answered most immediate questions. The only major problem was with the product details which were less well explained – the addition of more detailed information on which anti-virus engines, rules and so on are triggered, and why, would be a welcome addition.

Technical Details

All machines used in the test were identical 1.6 GHz Intel Pentium machines with 512 MB RAM, 20 GB dual hard disks, DVD/CD-Rom and 3.5-inch floppy drive.

Server software: *Windows 2000 Server Service Pack 2* with *Exchange 2000 Server Service pack 2* and *Outlook 98*.

Client software: *Windows XP Professional* with *Outlook Express*, *Windows XP Professional* with *Outlook 98*.

Developer: *GFI Software USA, Inc*., 201 Towerview Court, Cary, NC 27513, USA. Tel +1 888 2 GFIFAX; fax +1 919 388 562; email sales@gfiusa.com.

GFI Software Ltd UK, 5, Princeton Mews, 167–169 London Road, Kingston-upon-Thames, Surrey KT2 6PT, UK. Tel +44 870 770 5370; fax +44 870 770 5377; email sales@gfi.co.uk

Pricing: For pricing information see http://www.gfi.com/mailsecurity/msecpricelist.asp. Pricing starts at US$295 for ten mailboxes and includes virus updates for one year after purchase and three months of free support from date of purchase.

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, 6 Kimball Lane, Suite 400, Lynnfield, MA 01940, USA

Tel (781) 9731266, Fax (781) 9731267

END NOTES AND NEWS

The 12th International Virus Bulletin Conference takes place at the Hyatt Regency, New Orleans, LA, USA from 26–27 September 2002. Special *VB* subscriber rates apply. Contact us for more information: tel +44 1235 555139, or email VB2002@virusbtn.com. See the *VB* website for full conference programme details including paper abstracts: http://www.virusbtn.com/.

The International Symposium on Information Security 2002 will be held 1–3 October 2002 in London. Held in conjunction with the CISSP Annual Meeting, this event is a high-level forum for CISSPs and their information security associates. See http://www.misti.com/.

Black Hat Asia 2002 takes place at the Marina Mandarin Hotel, Singapore, 1–4 October 2002. Five training courses take place 1–2 October, with two tracks of presentations at the Briefings, 3–4 October. For further information see http://www.blackhat.com/.

Information Security Systems Europe 2002 will be held in Disneyland, Paris, from 2–4 October 2002. Presentations cover technology, infrastructure, applications, legal/political issues, and threats and responses. For more details see http://www.isse.org/.

The Third Annual RSA Conference 2002, Europe, is to take place 7–10 October 2002 at Le Palais des Congrès de Paris, France. As well as keynote presentations there will be more than 85 individual breakout sessions on topics ranging from enterprise security to hacking and intrusion forensics. A discounted registration rate applies until 16 September. See http://www.rsaconference.com/.

SANS Network Security takes place 18–25 October 2002 in Washington DC, USA. For details see http://www.sans.org/.

COMPSEC 2002 takes place 30 October – 1 November 2002 in London, UK. Presentations and workshops are held in four streams, covering management concerns, infrastructure, law and ethics, technical issues and case studies. See http://www.compsec2002.com/.

The CSI 29th Annual Computer Security Conference and Exhibition will be held 11–13 November 2002 in Chicago, IL, USA. The conference is aimed at anyone with responsibility for or interest in information and network security. For more information email csi@cmp.com or see http://www.gocsi.com/.

The 5th Anti-Virus Asia Researchers (AVAR) Conference takes place 21–22 November 2002 in Seoul, Korea. Topics covered will include information on how the AV community works together globally, the latest virus and AV technologies, and reports on virus prevalence in various countries in Asia. The conference will be hosted by *Ahnlab, Inc.* For more information see http://www.aavar.org/.

Infosecurity 2002 conference and exhibition will be held 10–12 December 2002 at the Jacob K. Javits Center, New York, USA. For further details, including information on exhibiting and conference registration, see http://www.infosecurityevent.com/.

The deadline for paper submissions for the RSA Conference 2003 is 16 September 2002. The conference will take place 13–17 April 2003 in San Francisco, USA. More information can be found at http://www.rsaconference.com/.

DialogueScience Inc*. has announced the beta release of *SpIDer Mail, allowing users of any email clients using POP3 to check incoming messages for viruses before they are processed by the mail client. The program runs under both *Windows 9x/Me* and *Windows NT/2000/XP* operating systems and is designed for virus protection of any client using POP3 to receive mail. *SpIDer Mail* is available for evaluation and testing, see http://www.dials.ru/english/.

***Websense Inc*. has announced the release of its *Premium Group III* (*PG III*) malicious websites database**. *Websense* imports millions of websites into its URL Warehouse and scans them for malicious code, including ActiveX controls, Visual Basic script, JavaScript, and Java applets. Then, using sophisticated algorithms, it identifies malicious code and enters those sites in the *PG III* database. The database is updated daily. Companies can then block employee access to those sites, preventing the subsequent download and execution of malicious code. See http://www.websense.com/.

The October 2002 edition of *Network Computing* magazine will be dedicated to security, with a CD cover mount containing up to 25 evaluation and demo copies of the security industry's most popular software or third-party applications. The issue itself will be the launch of *Network Security*, which will carry news stories, features, opinions and product reviews dedicated to computer security.