ISSN 0956-9979 DECEMBER 2001

VIRUS BULLETIN

THE INTERNATIONAL PUBLICATION ON COMPUTER VIRUS PREVENTION, RECOGNITION AND REMOVAL

IN THIS ISSUE:

• **The Early Bird:** *WormCatcher*, Roger Thompson's port 80 traffic monitor, played an important role in the discovery of later CodeRed variants and in the early alerting of the explosive spread of Nimda. Nick FitzGerald explains how *VB* readers may be able to help in the *WormCatcher* effort. See p.4.

• **Speaking in Code:** At the end of October, a Code of Conduct for Anti-Virus Professionals was released, intended to be signed, voluntarily, by those in the AV industry. Originator of the Code Kenneth Bechtel explains why he believes such a code is necessary and attempts to dispel some misunderstandings, starting on p.9. The Code can be found on p.8.

• **RAVe Review:** Marking *Virus Bulletin*'s first review of a *Linux*-based product, Matt Ham takes a look at *GeCAD*'s *RAV AntiVirus for Sendmail* and *RAV AntiVirus Desktop for Linux*. See p.18.

CONTENTS

COMMENT

“ Request a sample over the phone and have it mailed to your preferred Hotmail account – everything is possible these days! ”

Disturbing Trends – Can You Smell the Hype?

It seems that not much time has passed since a new virus alert was created and mailed to one of the many security mailing lists. According to the alert, a new worm may be circulating or arriving as an attachment with the extension .exe or .vbs. The new worm may be related to Bin Laden or the war in Afghanistan. It is reported to be Trojanized and will exploit your system and mail confidential information to other users.

Hold on a second – it was actually 24 hours ago that we received this report from one of our customers, who received it via one of the alert mailing lists. In fact, these types of report arrive on a daily basis and many of us spend countless hours trying to verify or counter their claims. It has become a disturbing trend to see so many security companies attempt to become the next *eEye* by releasing alerts that could match literally dozens of existing viruses or hoaxes.

Have you ever called one of these companies and requested additional information about an alert? I urge everyone to do this at least once. When I called the author of one of the recent media alerts, I was forwarded to half a dozen people until finally the so-called security company found someone who was aware of the press release. Claiming to be the person who initiated the alert, I was told that the information was based on a report from a ‘trusted’ third party. After a phone call to the ‘trusted’ third party (another e-security company), I learned that the alert was based on information from a newsgroup and a question from an Internet user. Trusted source, right?

Sad but true – we have to deal with this misinformation every day.

One may argue that it is even more disturbing how some of these companies handle malicious code. Without much effort, anyone can locate code samples for many of the recent worms and viruses on their Web sites. People unfamiliar with programming can access commented disassemblies and get a general idea of how to fix or improve the code. And you wonder why there is one new variant after another? The people telling us why we should be worried are the same people who have not learned that it may not be a good idea to mail virus executables to anyone that asks for them. Request a sample over the phone and have it mailed to your preferred Hotmail account – everything is possible these days! The last 12 months have taught me not to be surprised, even when the ‘senior researcher’ from an e-security company tells me that he accidentally ran the latest worm on his corporate computer and that he doesn’t know what to do next.

But let’s get back to the media alerts from the last 12 months. People would think this ill-advised hype only comes from e-security companies facing survival challenges. Unfortunately this type of behaviour is present also among established anti-virus companies.

Reported in the November issue of *Virus Bulletin*, numerous anti-virus vendors jumped on the bandwagon and alerted about the ‘Antrax’ virus, which turned out to be a non-working mass-mailer of the VBSWG construction kit. Not only did certain anti-virus vendors fail to test the virus prior to releasing an alert, they also failed to consider the impact a virus named ‘Antrax’ would have. Even worse, this was not the only time in 2001 where we had to advise customers not to worry about certain virus alerts. Equally unnecessary were the announcements of the ‘Jennifer Lopez worm’ (officially named VBS/Loveletter.CN), ‘WielkiBat’ (officially named VBS/VBSWG.AB), ‘LittleDavinia’ and many others. It seems one only has to look at the samples being posted to a particular urgent sample exchange forum.

Consider this for the future: before jumping on the bandwagon and making statements about a new virus (…or the end of the Internet), get your facts straight and consult with other researchers. There is no shame in asking for confirmation. Rather, it is a shame to rush to the media and cause all of us to waste our time.

Joe Hartmann, Trend Micro, USA

NEWS

VB 2002 Call For Papers

Virus Bulletin is currently seeking submissions from those wishing to present papers at VB 2002, in New Orleans, USA, on 26 and 27 September 2002. As in previous years, the conference will host two concurrent streams of sessions, corporate and technical. Abstracts of approximately 200 words must reach the Editor of *Virus Bulletin* by Friday 22 February 2002. Submissions received after this date will not be considered. Please send your abstracts (in ASCII or RTF format only) to editorial@virusbtn.com. Authors are advised in advance that the submission date for completed papers selected for the conference programme will be Friday 28 June 2002. For details of sponsorship opportunities at VB 2002, please email vb2002@virusbtn.com ▮

Bad Boys

British Internet Service Provider *BTOpenworld* has done more than provide Internet services for its customers this month. The ISP's customer support department has sent recent Badtrans variant W32/Badtrans.B to customers who had been seeking technical expertise from the department (as opposed to viral infection). Despite having no unique characteristics (the worm uses the same *Outlook Express* vulnerability exploit as Nimda) this worm has been spreading very rapidly. In fact, in the initial spread of Badtrans.B numbers rivalled those of SirCam – the first virus to do so in nearly four months – indicating the same old sorry tale that the message to install vulnerability patches and run up-to-date anti-virus software has yet to be heeded by the public at large ▮

Setting Standards

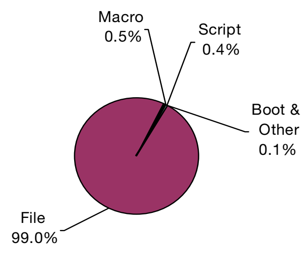
Industry standards seem to be the topic du jour, with *AVIEN* introducing its Code of Conduct for AV professionals (see p.8) and *Microsoft* announcing its intention to form an organization with security companies *@stake*, *Bindview*, *Foundstone*, *Guardent* and *Internet Security Systems* to create industry standards for handling security vulnerabilities. The as yet unnamed organization aims to develop security standards that are comprehensive, collaborative and broadly accepted. *Microsoft* acknowledges that the development and refinement of these standards will be a lengthy process and, in the short term, the members of the organization have made a pact to abide by some basic practices. The companies have agreed to report and address security vulnerabilities thoroughly and expeditiously, to allow a 30-day grace period before disclosing the details for exploiting the vulnerabilities and to exercise due diligence when developing security tools, to limit their use to lawful purposes only ▮

Prevalence Table – October 2001

Virus	Type	Incidents	Reports
Win32/SirCam	File	52650	92.7%
Win32/Magistr	File	1833	3.2%
Win32/Nimda	File	799	1.4%
Win32/Hybris	File	595	1.0%
Win32/MTX	File	126	0.2%
Laroux	Macro	92	0.2%
Haptime	Script	70	0.1%
Kak	Script	64	0.1%
Win32/BadTrans	File	38	0.1%
Solaris/Sadmind	File	35	0.1%
LoveLetter	Script	28	0.0%
Tam	Script	25	0.0%
Marker	Macro	24	0.0%
Ethan	Macro	22	0.0%
Divi	Macro	21	0.0%
Tristate	Macro	20	0.0%
Win32/Funlove	File	20	0.0%
VCX	Macro	19	0.0%
VBSWG	Script	18	0.0%
Win95/Spaces	File	18	0.0%
Win32/Apost	File	16	0.0%
Win32/Ska	File	15	0.0%
Class	Macro	13	0.0%
Win32/BleBla	File	12	0.0%
Win32/QAZ	File	12	0.0%
Win32/Bymer	File	11	0.0%
Win32/CodeRed	Other	11	0.0%
Win95/CIH	File	11	0.0%
Others [1]		195	0.3%
Total		**56813**	**100%**

[1] The Prevalence Table includes a total of 195 reports across 73 further viruses. Readers are reminded that a complete listing is posted at http://www.virusbtn.com/Prevalence/.

Distribution of virus types in reports



LETTERS

Dear Virus Bulletin …

'Tis the Season to Dump Software

It has been a bad year for *Microsoft*. The glitz and glamour of the *Windows XP* launch and Steve Ballmer's earlier MonkeyBoy dance can't hide the fact that they received a pasting from journalists and analysts alike as *Microsoft* security came under the spotlight. Recent reports suggest that, in the wake of Code Red and Nimda, a not insignificant percentage of their userbase have jumped ship from using *Microsoft IIS* as their Web server software.

The analysts at *Gartner Group* received much publicity by advising clients to dump *IIS* as soon as possible because of Code Red and adopt solutions with a better track record in security instead.

The refrain is heard time and time again: 'Your Web site got zapped by Nimda or Code Red? It's your fault for running *Microsoft IIS*!' To me this sounds disturbingly like blaming the victim rather than the culprit. 'You got mugged and your wallet was stolen? It's your fault for walking around after dark on the wrong side of town.'

Seeing as alternative Web server software has suffered from attacks in the last year as well (Sadmind and Lion come to mind), one wonders whether companies should spend their entire time hopping from one vendor to another as new attacks against their software are uncovered.

Indeed, if we follow the advice to its logical conclusion why isn't *Gartner* recommending customers dump *Windows NT* because of the high number of viruses that can infect that operating system? Well, because companies wouldn't consider it for a second as an appropriate response. And a knee-jerk reaction of removing *Microsoft IIS* instantly from your Web servers isn't an appropriate response either.

Although *Apache* and other Web server software does seem to have a stronger track record when it comes to security than the lads in Seattle, firms need to think very carefully before throwing the baby out with the bath water.

Ripping out your existing IT infrastructure because of Nimda and Code Red may do more harm than good. A more considered response, investigating why vulnerability patches were not put in place in good time, for instance, may bring to light underlying and more subtle problems with the way in which your company handles security.

Graham Cluley
Sophos Anti-Virus
UK

SHOWCASE

WormCatcher

Nick FitzGerald
Computer Virus Consulting Ltd, New Zealand

During the closing Q&A session at VB2001 in Prague, and on several other occasions during the conference, reference was made to something named '*WormCatcher*'. Unlike many other product references from users, mentions of this product were all positive, highlighting its role in the discovery of later CodeRed variants and in the early alerting of the explosive spread of Nimda. During the closing session of the conference I commented that *WormCatcher* was in need of more European reporting sites and that it would soon be 'going public'. This is a brief introduction to *WormCatcher* and explains how some of *VB's* readers may be able to help.

Seeing Red

Following the initial outbreak and discovery of CodeRed, the second variant – the one with the fixed random IP address generator, *not* the one that calls itself CodeRedII – was released (I shall refer to these variants as CodeRed.A, CodeRed.B and CodeRed.C, respectively). While tracking these developments, it became apparent that the traditional computer security community was not terribly concerned with 'exact identification' issues, nor even with moderately precise identification. Following its discovery and analysis, the developers of most (network) intrusion detection systems (NIDS or IDS) were quick to add detection of CodeRed. However, most of these detection 'signatures' were very generic, typically detecting only the initial buffer overflow part of the .ida request. One signature posted for a very popular freeware NIDS detected simply the string '.ida' in any traffic on any port!

This lack of precision, and apparent lack of concern about it, was quite obvious to me around 19–20 July, when CodeRed reports went ballistic. The first full CodeRed sample I received that had been captured in a network trap turned out to be what we now know as CodeRed.B. At that point the only detailed analysis of CodeRed was *eEye's* disassembly and the accompanying description. However, it was clear from our understanding of CodeRed.A that the sudden, dramatic increase in CodeRed reports from all round the Net did not fit with what the code should do.

On dumping the binary from *eEye's* disassembly and comparing that with the sample I had received, it was clear that 15 bytes were different between the two. Part of *eEye's* disassembly was commented simply as 'padding' and there was some self-modifying code whose role was poorly understood at the time. Further, the worm sent a copy of itself from memory, so some minor differences, such as the next victim address, were to be expected. Six of these 15 different bytes were outside the ‘padding’ area – some altered instruction parameters, some changed opcodes – yet few, if any, NIDS were reporting this as anything other than ‘CodeRed’. To most of the world it seemed as if nothing but the frequency of CodeRed reports had changed.

Very quickly it became clear that these minor code changes represented a new variant, CodeRed.B, whose functional random address generator explained the rapid increase in CodeRed detections. It seemed rather important to me, and many others in the AV sphere, that such differences should be detected, but this was not helped by the way typical NIDS detected CodeRed nor by the fact that non-vulnerable Web servers log only the GET request itself.

The GET request contains the buffer overflow CodeRed depends on to gain control [see *VB* August and September 2001 for details], but this could be the same in different variants of the worm (as it was in CodeRed.A and .B). Failure to log the request body meant that the worm body – the code that does the real work – was not logged so differences in it were unlikely to be detected through log file inspection. Further, vulnerable machines fail to log anything, as the overflow and instantiation of the worm never returns control to the process handling its GET request and *IIS* does not log anything until the process request returns.

Rose-tinted Spectacles

Despite the fundamental difference in the spread rate of these two CodeRed variants, in general the computer security community lumped all CodeRed reports together and seemed unconcerned with separating them. The arrival of CodeRed.C, with its more extensive compromise of victim machines and ‘improved’ target address selection algorithm, made differentiating reports by variant seem even more important (at least for those within the AV community).

In mid-August Roger Thompson, director of malware research at *TruSecure Corporation*, contacted some fellow AV researchers and asked if they would be interested in running a port 80 (HTTP) traffic monitor he had written. Dubbed *‘WormCatcher’*, this monitor acted as a trivial (in the best sense of the word!) Web server, listening on port 80 and logging whatever was thrown at the port. It distinguished between the then three known CodeRed variants and several ‘standard but innocuous’ port 80 happenings. Just as important, however, it raised alerts if it received any ‘unknown’ traffic on port 80. In the event that anything unknown arrived at port 80, event details (such as sending IP address and the time and date) were logged, the traffic written to a file, and the file emailed to Roger in case further analysis was warranted.

During this initial usage phase, *WormCatcher* was trained quickly to handle a whole host of relatively innocuous, but common, port 80 events and Roger added many program options at the suggestion of test users. Now *WormCatcher* typically runs at startup and minimizes to an icon in the system tray. It allows the user to specify an email address to copy its reports to (usually the user’s address), has options to draw the user’s attention to detection of ‘new’ things such as animating and changing the colour of the system tray icon or playing a sound file, and the port it listens on can be changed.

At this stage in its development, *WormCatcher* seemed pretty much ‘done’ – ready for wider deployment. During its testing and development period, it had caught very early samples of CodeRed.D and highlighted some ‘problems’ with proxy servers trashing many CodeRed spread attempts (this last finding alone throws a substantial amount of NIDS-derived CodeRed data into question and tempers many conclusions that some have drawn from such data).

Nimda, Nimda, Nimda …

On 18 September, things changed. Win32/Nimda.A@mm was detected in vast numbers, spreading quickly via multiple vectors. One of these was via *IIS* exploits which started flooding the Net, and hence *WormCatcher* test sites around the globe. A simple form of rate-limiting was included in *WormCatcher* so the standard reporting address would not be too badly flooded in just such an event. However, the fact that Nimda spreads its code by means other than directly through its worm-like process meant that *WormCatcher* needed some changes to improve its variant detection and reporting capabilities – its *raison d’être*.

These improvements have been made and the *WormCatcher* network is looking to add some more sites, particularly in Europe. The requirements to run a *WormCatcher* are quite simple: a Win32 machine (*Windows 9x*, *NT* and *2000* have been tested and *Windows XP* should work) that is not already running a Web server (or other service on port 80); that is visible to the Internet on port 80 (you may have to open a specific hole in your network firewall); that has access to an SMTP server so traffic reports and samples of ‘unknown traffic’ can be sent and that is (almost) always on is a big plus. Finally, a willingness and policy approval to provide to an outside agency data about some of the Internet traffic that reaches your network. If you are interested in joining the *WormCatcher* effort, the latest public release can be downloaded from the Web site http://www.wormwatch.org/ and Roger Thompson is happy to answer email queries (email rogert@mindspring.com).

The Future

Currently Roger is working on processing *WormCatcher* traffic reports automatically with the aim of posting detection statistics on the Web site. A likely feature addition will be automatic polling of the *WormWatch* site for updates to the global detection file. Among other things, this will allow newly isolated worms to be identified by their correct names, rather than by the rather odd temporary names they are given after being added to the local detection file.

VIRUS ANALYSIS

Remote Vision

Costin Raiu
Kaspersky Lab, Romania

It has become almost habit that whenever a new vulnerability is discovered in some software or an operating system and announced, be it in a *Microsoft* product, some flavour of Unix or a third-party software package, an Internet worm is written to exploit and propagate through the respective vulnerability.

At first, this was a domain exclusively reserved for Unix malware, but it was changed forever by the appearance of CodeRed and Nimda, which exploit various bugs specific to *IIS* running on *MS Windows* platforms. Even though, from the point of view of spreading, these two worms are between the highest ranking of all time in their class, from time to time highly-successful worms appear for other operating systems, such as *Linux* or *FreeBSD*.

Occasionally, less-successful worms appear which, thanks to the prompt action of experts in the security field, are stopped before they manage to hit the Internet at their full strength. The *BSD* worm 'ExSee' is an example of such a worm.

A Buffer Overflow Exploit

Around the middle of June 2001, the 'TESO Security Team' – a name which may sound familiar to some (see *VB* June 2001 p.6) – discovered a security problem which affects most of the Unix distributions based on the *BSD* source. An unchecked buffer operation allows a malicious attacker to trash the stack of the 'telnet' daemon process, possibly resulting in execution of infiltrated code with high (usually root) privileges.

Even though the details of the exploit were not made public at that time, about one month later, a demonstration sample made its way to the Bugtraq discussion list, becoming available to anyone interested in acquiring a copy of it. The post triggered a prompt reaction from the author of the exploit. The author pointed out a specific note in the exploit code which stated that distribution to third parties was specifically prohibited, particularly distribution through the Bugtraq mailing list, or a specific Web site which is known to host various exploits and security tools. Unfortunately, by this stage it was too late, and two months later a worm using the telnetd buffer overflow code from the aforementioned sample was reported.

The Worm

Initially called 'x.c', or 'ExSee', the worm is about 6 KB in C source code, and uses no additional or external scripts.

An interesting thing about this worm is that, whenever it replicates to a remote system, it tries to run a couple of commands on the target machine, one of which is a command to download a copy of the source code from a site in Poland, 'mri.am.lublin.pl'. Apparently, the site has nothing to do with the worm – I assume the author hacked the site and used it to host the source which is downloaded, compiled and run on infected machines.

Fortunately, the copy of the worm source was removed from the Polish site soon after the worm was discovered, meaning that the epidemic went almost unnoticed.

However, an open remote shell backdoor is installed on infected machines, which listens for connections on port 145, the so-called 'UAAC Protocol' port. Despite the fact that the worm is no longer available for download from the site, and even though it is not carrying the infection further, an attacked system will have an open root shell on port 145 – which is dangerous enough by itself.

Propagation

Whenever the worm finds a vulnerable system, the exploit code is sent, along with a series of commands designed to propagate the infection further.

First, as mentioned above, the worm attempts to download a copy of itself using the classic *BSD* tool 'fetch', and save it into the system root folder. Next, the worm compiles itself, and deletes the source saved in the previous step. It will remove all debug information from the compiled executable, making it no larger than about 6 KB, and move it to the /usr/sbin directory under the name 'cron ' (note the space at the end of the file name). Then, the worm executes itself on the remote system, and ensures that it will be started every time the system reboots from the system script /etc/rc.local.

As mentioned above, the worm also backdoors the system by adding a line at the end of the /etc/inetd.conf file, making sure to restart the Internet services daemon, which activates the backdoor.

Replication

The worm's replication subroutine targets random IP addresses across the Internet. The random IP generation uses the built-in C random functions, and seeds the generator with the current time. This allows the worm to hit a rather large number of IP addresses, but it is not as efficient as other modern worms which allocate higher probabilities to IP addresses that are similar to that of the hosting system.

In addition, the replication process is slowed by the fact that the worm waits ten seconds before sending the exploit code, and pauses for another second before sending the commands to the infected system.

Given the high probability that a random IP address is unused, the timeouts will slow the replication a great deal. Moreover, the exploit requires a very large amount of data to be sent to the attacked machine, which is time-consuming in itself.

These factors, combined with the missing source from the download location, prevented the worm from becoming widespread.

However, it is interesting to note that between September 12 and the time of writing this article, Smallpot – a custom honeypot software I have designed, which collects hacking attempts and various other data sent to it – received three attempts to check for the vulnerability used by ExSee. These were from three different IP addresses, one of which connects to a *NetBSD* machine.

The variations in the number of port 23 (telnet) network scans can be investigated on sites such as Dshield (http://www.dshield.org/) – as usual in such cases, it's impossible to state for certain whether the port 23 connection attempts are caused by the worm, or by hackers attempting to find vulnerable hosts. Given the reasons explained in this article, it is my belief that the former is a much more reasonable explanation.

Technical Details of the Exploit

Before attempting to send the exploit code to a remote host, the worm performs a couple of checks. First, even if it managed to initiate a TCP handshake for port 23 with the remote host, it will try to read some data from the socket, which should be a traditional Unix login banner. This makes sure the remote system is at least attempting to communicate on port 23, and it's not a dead end.

Next, the worm sends a specially-crafted test string, which attempts to query the remote telnet server for several supported options. This string, in hexadecimal form, is as follows: 'FF F6 FF FB 08 FF FB 26'. 'FF' is a special marker character in the telnet protocol, which indicates that a control code is coming. 'F6' is an 'are you there?' code, meant to query the remote party's ability to understand the telnet protocol codes. 'FB' is a control code which announces the client's desire to negotiate some specific option, in our case '08', which is output line width and '26', which is encryption.

The worm expects the server to answer with a specific string, which should be: '0D 0A 5B 59 65 73 5D 0D 0A FF FE 08 FF FD 26 00'. This translates as 'CRLF', '[Yes]', 'CRLF', and responds to the negotiation request – 'no' for output line width and 'yes' for encryption. Also, the worm checks whether the server has replied with more information regarding the encryption protocol, which indicates the availability of various encryption interfaces (Kerberos, etc.).

Depending on the extended answer from the server, the worm uses different exploit data to construct the strings which trigger the buffer overflow. Next, the worm attempts to set a massive 31,500 environment user variables to its internal exploit code, padded with x86 NOP instructions, to increase the odds of the IP hitting the shell code after the stack overflow.

If the operations (which are likely to take a very long time – particularly on a slow link) are successful, a 250-byte-long telnet protocol command string is sent, which eventually causes the stack overflow and the return in the memory area holding one of the many user variables with the shell code. The shell code calls the execute process system API, asking it to run the traditional Bourne shell, '/bin/sh'. From here, the attacking copy of the worm runs its replication commands.

Conclusions

Even though the 'ExSee' worm was unsuccessful, it remains as another warning that new bugs found in various software programs or operating systems can be exploited through 'mobile' code, making them even more dangerous as they have the potential to target more computer systems. The slow replication algorithm of this worm, together with its random attack pattern, decreased its chances of making it to the wild, but the removal of the source code from the remote site was also a very important step.

However, the main point is that, with more and more malware attacking Unix users, the AV community will have to start building links with the security people working natively in this field. Additionally, AV vendors may need to take steps to increase the protection of their data security packages with Intrusion Detection Systems. In this case, for instance, an anti-virus product is not able to prevent ExSee from installing the backdoor in the attacked system. However, an IDS using updated definitions should easily be able to block the exploit from taking place, thus preventing the malicious code from being executed.

Name: BSD86/ExSee.A

Aliases:	'x.c', BSD/Walk.worm.
Type:	Network-propagated worm affecting x86 BSD-based Unix systems.
Payload:	Backdoors the infected system by adding a root shell on port 145.
Detection & disinfection	Infected systems have a copy of the worm stored in a file named '/usr/sbin/cron\x20'. Delete this file, patch /etc/rc.local to remove the worm link, edit /etc/inetd.conf and remove the bogus UAAC service. Then download and install a patch for the telnetd exploit from your vendor.

SPOTLIGHT

In late October 2001, the Anti-Virus Information Exchange Network (AVIEN) announced the general release of the Anti-Virus Professional's Code of Conduct. The Code is aimed at all those in the anti-virus industry as well as general security professionals worldwide. Hosted on the AVIEN Web site (http://www.avien.org/codeconduct.html), the intention is that the Code be printed out, signed in the presence of witnesses and faxed to a representative of AVIEN who will maintain the database of signatories. Thus far the Code has generated a great deal of debate. The following is the Code in its current form, and in the feature starting on p.9 Kenneth Bechtel, the Code's originator, explains the thinking behind it.

Anti-Virus Information Exchange Network Code of Conduct

(i) DO NO HARM

I will not write and deliberately release any code with malicious intent. With malicious code being defined as not only code that does direct or indirect damage to systems and data, but also code that has undesirable secondary consequences such as risk of embarrassment to or punishment of the victim.

I will not write replicative or destructive code unless I am convinced that it is necessary for internal research or testing purposes as required and defined by my professional activities. If I regard it as necessary to write such code, I will do so under secure and strictly controlled conditions, and I will not publish such code. Nor will I share it unless it is absolutely necessary, and then only with individuals whose competence and adherence to this code of conduct or an equivalent is beyond question. I will not keep copies of such code for any longer than is strictly necessary, and only under secure and strictly controlled conditions.

I will not deliberately damage live data. Nor will I alter any data except as authorized by the owner of those data.

I acknowledge that the public release of Malware, even for benevolent purposes such as advising potential victims of vulnerabilities in their systems, is never beneficial if it involves unauthorized access or modification to systems, even if the quality and safety in use of the code could be guaranteed under all circumstances.

(ii) DUTY OF CONFIDENCE

I will treat as confidential all data entrusted to my care. I will not divulge my client or employer's identification, or claim to act as their representative, except with their expressed consent, or where an overriding legal or moral obligation exists.

(iii) DUTY TO BEHAVE RESPONSIBLY

I will behave at all times in accordance with all applicable laws, policies, and codes of conduct required by AVIEN and any other organization with which I am affiliated.

Other than for publicly accepted legitimate development or research as part of my professional activities in understanding and/or creating defenses against malware, I will not intentionally trade, solicit, or transmit malware, or encourage these activities. I will always discourage such activities other than for publicly acceptable legitimate development, testing or research. I will not pass on malicious code to anyone whose competence and integrity is in doubt.

(iv) DUTY OF CARE

Malware entrusted to me in my professional capacity will be handled with the utmost care and respect for their capabilities for harm, in order to prevent infection or dissemination.

I will assume responsibility for viral incidents when charged with their management, irrespective of whether they result from any action of mine.

If contacted with details of a possible infection, I will proceed as if there is a definite, proven infection until it can be proved otherwise. If any system in my charge is infected, I will advise all individuals or organizations who may have been a source of infection, or who may have received malicious code as a result of contact with those systems.

(v) DUTY TO INFORM AND EDUCATE

I will dispel Malware hype, myths and misinformation through education. I will not claim knowledge or ability beyond my actual capabilities. I will not use Malware-related hype or fear-mongering to promote any company, any product, or myself.

I acknowledge and recognize that Virus eXchange (vX) web sites and bulletin boards only further the malware problem. I will not validate their existence by frequenting them, other than for ethically acceptable research into their activities. When asked, I will support and assist authorities in discouraging and suppressing vX activity wherever possible.

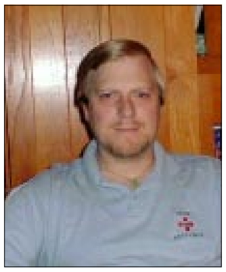
I understand and agree to this Code of Conduct and pledge to act in an ethical and professional manner, as outlined above.

By signing this document, I agree to abide by any reasonable penalty imposed by the AVIEN appointed controlling committee, if found guilty of unprofessional conduct in breach of this Code of Conduct.

FEATURE

Tilting at Windmills

Kenneth Bechtel
Team Anti-Virus, USA



I have to admit that, when I started working on developing my concept of an industry-accepted Professional Code of Conduct, I expected resistance. However, the reception the Code has received to date cannot be elevated even that high. The most common questions I have been asked are, in fact, the ones that I started out with myself: is a Code of Conduct needed, and why? I hope to answer these and a number of other questions and concerns in this article.

Building a Code of Conduct

The question of ethics in computer virus research can be traced back to the earliest days of this becoming a career field. Evidence can be seen in the rules for Virus-L, where, for both security and bandwidth reasons, virus code was not permitted. Small parts of disassemblies were encouraged for discussion, but full source code was not.

The *Computer Anti-virus Researchers' Organization* (*CARO*) established a set of rules for its membership, as has every organization that has been set up since. However, a while ago I noticed that many industry pundits were pointing out that, unlike many other professions, the Information Security field has no common code of ethics or conduct. At the time I began a rough draft of a code of conduct and since then have pursued many organizations to pick up the challenge.

After several years, I recognized where, when and whom I should approach. In the early days after the *Anti-Virus Information Exchange Network* (*AVIEN*) was formed, I put two questions to the members. One concerned a code of conduct; the other will be saved for a future article. Over time a small committee formed which developed the Code of Conduct that was released to the public on Monday 29 October 2001.

Defining, Refining and Redefining

The Code of Conduct committee took the original framework of the Code and started refining it. Initially, there were many areas that were defined strongly along the lines of 'Thou shalt' and 'Thou shalt not'; it was very black and white. During discussions, comments such as 'I don't like the wording of this' were raised frequently in some sections of the Code, and in others objections were raised such as, 'What about this situation? – surely that's an acceptable behaviour, and should be an exception.'

I must admit to having felt very discouraged, and on more than one occasion, stated, 'I am confident that the Hippocratic Oath could not be written in this day and age.' It took the committee at least seven revisions of the Code, and one false start, before we came up with a document that was felt to be strong enough, and yet provide people sufficient flexibility to do their jobs, and which we felt we could release for the world to see.

The idea of breaking down the Code into several smaller, more targeted codes was suggested many times. However, I felt and still feel this would be taking our eyes off the goal of a unified, commonsense professional code of conduct for the industry.

By establishing this Code of Conduct we are presenting a global front, in a similar way to that in which all lawyers and doctors are seen to abide by a code of ethics. Many of us in the industry operate on a very similar personal code already. This is demonstrated by the various organizations that have made their code public – *WildList*, *EICAR* and *AAVAR*, to name just a few.

What we have attempted to do is to take existing ideas, and produce a single code to show to the outside world. We are the people who are putting it into practice on a daily basis, we are not lawyers, and we did not write this with a legal perspective (although sometimes I wonder whether a lawyer should have been involved).

What are the Benefits?

Another question that I have been asked many times since the Code was drafted is 'what are the benefits of signing the Code?' Well, by signing the Code you will not become rich, nor more popular with the opposite sex. However, you will be making a statement to the outside world showing that you are ethical and that you have agreed, *voluntarily,* to abide by a set standard. This comes in handy for the corporate supporters who are questioned frequently by management on matters such as ethics. For those employed by vendors this gives your employer the ability to say 'We are good corporate citizens and our employees are encouraged to sign the industry standard.'

In its role as sponsor of the Code of Conduct, *AVIEN* has offered Web space for a 'Who's signed the Code of Conduct' Web page. On this page, those signatories who wish to be listed publicly, will be listed by name and corporate affiliation. This isn't a perfect solution, but is seen as a good starting point. In addition, *AVIEN* will maintain, offline, more detailed personal information about signatories (in order to help with enquiries such as 'is the Ken Bechtel

I'm talking to the same Ken Bechtel who is listed on the public Web page?').

A number of individuals have questioned the wording of the Code. We have done our best to anticipate all the situations that are in the best interests of the end user community. The Code is not intended to replace your common sense, nor does it need to be 'read into' to interpret our intentions. It is a face-value document. There will be people who try to read into it other meanings. For instance; there have been comments that the act of deleting your own data breaks statements such as 'I will not deliberately damage live data'. This sort of statement is not designed for that, but is intended to be a very commonsense thing.

Put another way, 'I will not deliberately damage live data' implies 'I will not release a virus on a production network', 'I will not modify another's works without their permission', 'I will not delete or modify files with malicious intentions in networks and systems not deliberately set up and designed for that purpose'. This statement does not preclude a researcher from infecting goats in a lab setting, and I think it's pretty straightforward in prohibiting things like revenge Trojans, or other acts, without impacting on the way we perform our jobs.

Why AVIEN?

Why was *AVIEN* chosen to sponsor the Code? Well that's probably the easiest, if least popular question to answer. It can be summed up in two words: vendor neutral. Of all the organizations devoted to fighting computer viruses, *AVIEN* is the only one that is truly neutral and has no vendor representatives in positions of influence or in its members-only discussion lists. Vendors are welcome to participate in *AVIEN*'s Early Warning System (EWS) functions and lists, and have a voice there should they desire, but as far as organizational structure, discipline or membership issues are concerned, vendors are notably absent.

The reason this is an important issue is the lack of politics. Furthermore, *AVIEN* as an organization does have a Disciplinary Committee, which became important when we stopped to consider what we would do should someone violate the Code. Finally, even though *AVIEN* is the sponsor, it will receive no direct revenue from the Code, which is freely available for anyone to sign.

Enforcement

One of the biggest debates we had while drawing up the Code of Conduct was what to do with Code violators. We started with a simple idea (OK, it was my suggestion): if we are to reward signatories by posting their names and affiliations publicly, why not expose those who violate on another page? That idea was received with a notable lack of enthusiasm!

In the end it was decided that, since it is probable that there will be different degrees of violation, and since *AVIEN* is the sponsor, *AVIEN*'s Disciplinary Committee should be left to tailor the punishment to fit the crime. That way if, after investigation, it is determined that although a violation occurred it was without intent, we could censure the individual. While, if it was found to be a flagrant, deliberate violation of the Code (along the lines of 'you fired me, now kiss your database goodbye'), we could take more drastic actions.

Misunderstandings

There have been several minor misunderstandings concerning the Code. The most prevalent of these is that those who sign the Code of Conduct are obliged to obey the rules of *AVIEN*. This is *far* from the truth: when people sign the Code of Conduct, they are agreeing only to abide by the Code and, should they violate it, abide by the reasonable punishment imposed by the *AVIEN* Disciplinary Committee.

Another misconception is that this is a closed copy. While the Code of Conduct committee has worked long and hard, we recognize that the face of the Anti-Virus industry changes almost on a daily basis. While the basic intentions of the Code will never change, we will need to keep it fluid and evolving.

We also recognize that, while we were the authors of the Code, in order to gain acceptance by the widest number of professionals, it has to be accepted by them, and we need to listen to their input. The Code will change, and currently we are on a three-month review cycle. We don't want to review it more frequently than that, because those who have signed may not like having to re-read new wording every day, week or month. Three months was deemed a reasonable period between releases.

The Way Forward

In the global society today, mature, established, professional fields have codes of conduct by which all practitioners of the given profession abide. This is true from the military, to lawyers and the medical field. In some cases, there are more detailed and restrictive codes depending on your position within the field, but there is always one base code, by which all members are expected to abide.

There is no such thing as a perfect code of conduct, or ethics. Codes are expected to be frameworks that define what members of a profession expect from its practitioners. The anti-virus profession has long been small, and in some eyes exclusive. It's high time that politics, egos and agendas were put on hold and we present a united front to the outside world and show them that we have standards and hold our responsibilities to the user community as a high calling.

I welcome any comments and feedback on the Code via the editor of *Virus Bulletin* (email editor@virusbtn.com). Let's make a great thing better.

OPINION

Chopping Off the Tail – Revisited

Peter Morley
NAI, UK

[*Last year, Peter Morley started a discussion of how he felt the efficiency of anti-virus software could be maintained by removing detection and repair of old, now inactive malware (see* VB *September 2000 p.8 and November 2000 p.11). This month he returns with an update on the situation, and proposes some changes for the coming year - Ed.*]

Before I begin, let me say that we are discussing reducing detection of legacy viruses (old-fashioned DOS file viruses) only. Nothing in this article is about Boot sector viruses, multipartite viruses, *Windows* viruses, high-level language viruses, macro viruses, script viruses, Bat file viruses, Internet worms, password stealers, or Trojans.

Since my previous articles are now over 12 months old, I shall start with a summary of the progress since those were written – where we are now, and how we got here.

Where We are Now

i) The numbers game

At the time of writing (October 2001), we claim to detect 58,978 viruses, trojans and variants, of which 32,651 are legacy. Both figures are gross underestimates, because we detect and repair thousands of items we have never seen, using generic techniques. When we get such an item, all we do is check the repair. We do not add it to the count.

So, some 50% of database size and scan time are wasted, scanning for items in which no one is interested.

ii) Inputs

Currently, we swap viruses monthly with up to 14 other vendors. The legacy virus input in these collections has fallen from over 35 work pages 18 months ago, to under two work pages now. I believe it will reach zero within the next 18 months.

Those legacy viruses which do still appear are written mainly in Korea, China, Russia, and South America. We process them in full, droppers, rubbish and all, just as we always did, because reviewers get them too, and may well creep them into a test.

iii) Support calls

The prevalence table in *Virus Bulletin* has stopped showing legacy viruses. I can confirm that our own support units are not receiving calls about them.

iv) Reviewers

All reviewers responded positively to my request to stop using legacy viruses in test suites, although (as expected) they all reserved the right to use any legacy viruses they considered 'important'. Several of them are moving towards Proactive Testing (testing against future viruses), and in that case the problem goes away.

This last comment does not include *VB*, but they are not a problem anyway, because they make listings of their test viruses available, and they avoid putting rubbish in.

v) The Misery Test

The Misery Test (which demonstrates that the engine works if we double the number of viruses), still works fine, so the pressure to improve by removing detection is reduced.

The Last 12 Months

I said I had removed detection of Trivial.18, and awaited the holocaust. There was not a single comment.

I have removed detection of all non-virus generators, and the non-viruses they generate. Although there has been no comment, we did receive two new generators we had not seen before. Of course we now detect these, in case reviewers use them.

I have removed detection of most of the items which ask whether they should infect, before they do. There has been no comment about this either.

I have removed detection of most of the 'appendeds' which still run, and can't infect anything.

I have removed detection of some 'old rubbish' (with emphasis on the word 'old'). For eight years I have been putting detection of rubbish in, because it is easier to do so than to argue about it. Again, there has been no comment.

I took my life in my hands, and removed detection of some old droppers (again, with emphasis on the word 'old'). Although there has been no external comment, this has made management rather nervous. They are particularly nervous about the effect on large customers who keep old collections and run tests against them. I'm nervous too, because I know that if we get one of these we may well mishandle it.

The correct way to deal with this is to say ‘We removed it deliberately, because we thought it was not important. If you would like us to put it back we will, for three years, or forever (you choose). But when do you think we could reasonably take it out?’

Where do we go from here?

This time I’m *not* ducking the issue of removing detection of viruses, as I did in my two previous articles. So here’s what I want to do.

i) The numbers game

Obviously, we cannot just remove viruses slowly, because the virus count would fall slowly, and that would give rise to a stream of queries. So it is vital to put in place first a counting system in order to avoid this problem. What I propose is this:

Change the count of legacy viruses detected, from counting viruses and variants to counting detection and repair categories as one each.

This would reduce the count of legacy viruses from the 32,651 mentioned previously to 2664, a reduction of almost 30,000. It would replace the long slow fall, with a quick large fall.

Clearly, this cannot be done until sales, marketing and support departments are all fully briefed and well prepared. The timing of this operation needs to be agreed.

If we go ahead with this, I expect it to happen approximately second quarter, 2002. But we may not proceed, due to the ‘cold feet’ problem.

Incidentally, the change would provide an excellent marketing opportunity, but one which would require some groundwork. It must include preparation to handle those big customers who keep internal collections, as I mentioned earlier.

We will need to stress that the change of count does not, at this stage, include a change in detection capability.

ii) Removing the code

Three to six months after we make the counting change, we will be in a position to remove detection as we please, provided we choose carefully what to take out – so as not to cause bad reviews, or a deluge of support calls. The big generics (which will be counting as one each) can be left to the end. The additional slow reduction in count will not be noticeable, as it will be more than compensated for by later virus types being added.

I believe the process of removing detection should start sometime between mid-2002 and mid-2003, and that it will continue for three to five years. Some items (such as the historical classics Jerusalem and Cascade) will probably never be removed.

FEATURE SERIES 1

Worming the Internet Part 3

Katrin Tocheva
F-Secure Corporation, Finland

[*Recent years have seen a dramatic increase in the number of worms that spread via the Internet, and in particular the number of worms that use email clients. In this series Katrin Tocheva has looked at the spread of worms using the Sendkeys and CreateObject functions to target Microsoft applications and in this, the concluding part, she discusses worms that use different email clients and worms that target other types of application - Ed.*]

Pegasus Mass-Mailing Methods

A small number of viruses spread using the *Pegasus* email application – examples include HLLP.Toadie@mm, W97M/Jim@mm and W97M/Moridin@mm.

Toadie uses a spreading method in which it replaces the content of outgoing emails with its own message. It attaches the infected executable file to the email and sends the virus’ message instead of the user’s message.

Jim and Moridin use a similar method. First, they locate the *Pegasus* installation directory, searching in four possible default folders. Once it is found, they search in this folder and its subfolders for .pmw (ready to send) files. Then, using the information for the original recipient and email subject, they append the necessary fields, so the virus will send its own body text as well as the active infected document as an attachment (AT:).

A bug in Jim replaces one wrong line in the .pmw file and, as a result, the ‘To:’ field is missing in the infected message, thus making the spreading routine obvious. In Moridin this bug is fixed and the virus spreads. However, since *Pegasus* is not a widely-used email client, this spreading method does not pose a great threat.

Eudora Mass-Mailing Methods

There are two methods of spreading that use the *Eudora* email client.

i) Create a Message File

W32/Sysclock uses this method. First it attempts to locate the *Eudora* installation folder using the *Windows* registry (Software\Qualcomm\Eudora\CommandLine). Once this location has been found the worm collects email addresses by searching in the Out.mbx file – *Eudora*’s outgoing mail folder. Next, the worm creates its own .msg file (User.msg), adds to it the email addresses it has collected, then builds a message subject and body and attaches its code (X-Attachments: c:\pkzip.exe). To execute this .msg file and send itself via email, the worm activates *Eudora* sendmail.

This method of searching for email addresses has the potential to result in many more recipients than those methods that collect addresses from the address book – if, for example, the outgoing mail folder has not been cleared for a long time.

ii) Virus Adds Itself to the Outgoing Folder

Redteam virus uses this method. To achieve rapid mass-mailing it sends itself to all email addresses listed in *Eudora* nicknames file Nndbase.toc. This is the file that contains the aliases of the email recipients stored in *Eudora*'s address book. Redteam adds its own message to the outgoing *Eudora* folder (Out.mbx file) and changes the table of contents file (Out.toc) so that the worm will be sent out.

The infected message contains the virus code as an attachment. The subject and body text of the infected message make clever use of social engineering to entice the recipient to launch the attachment.

SMTP Methods

There are two main SMTP methods of spreading. The first patches the original WSock32.DLL and is used by W32/Ska. The second is the method that was used by recent widespread viruses Sircam and Nimda, which each use their own SMTP engines.

i) Patching the Original WSock.DLL

The first thing a worm such as W32/Ska does is create a copy of itself (in this case, Happy99.exe as Ska.exe in the *Windows* System directory). Then the worm saves the original *Windows* socket library WSock32.DLL (copied as WSock32.Ska in this example). Next it patches the original WSock32.DLL in order to intercept postings from the infected user (via email and/or to newsgroups). This means that the worm has control over the network activity and particularly over postings via the network that are made from the infected user.

In this case, the user's original email message (or posting to a newsgroup) is sent in addition to Ska's message, which contains the headers of the original posting and is made to look as if it comes from the same user. Ska's message contains an additional header 'X-Spanska' that is invisible to most users [P. Ször]. The worm file is attached, uuencoded, to this posting.

Since the discovery of Ska in the Wild, other worms have been developed which use the same method to patch the original WSock32.DLL. These include W32/MTX, W95/Android, W95/Suppl, and W95/Babylonia. Babylonia differs in that, instead of creating its own messages, it attaches a copy of itself to each message that is sent from the infected machine. It does this by patching WSock32.DLL (by replacing the 'Send' function with its own.

This method of spreading is slow mass-mailing and, since these worms require human intervention to open the infected file attachment, they are classed as chain letters.

ii) Using a Built-in SMTP Engine

This method can be split into two groups; methods in which spreading depends on the user's email settings and those that are independent of the email settings.

Sircam belongs to the group in which spreading depends on the user's email settings – while it uses its own SMTP engine, it will not spread if the user settings are not set appropriately.

Fast mass-mailers such as Nimda and Klez fall into the group whose spreading is independent of the local email settings. They connect directly to the SMTP server of the destination domain.

Nimda's mass-mailing spreading method uses SMTP protocol to connect to the email server of the destination domain [G. Erdelyi 1]. The worm collects email addresses and builds a list of recipients. Using MAPI it collects the senders' email addresses found in the Inbox. In addition, it collects the email addresses found in .htm and .html files that are located in the user's Temporary Internet Files folder.

Once Nimda has prepared the list of email recipients, it sends email messages to each of them. It extracts a MIME message from its body and appends its MIME-encoded copy to it.

Nimda uses an exploit in *Internet Explorer*, *Outlook* and *Outlook Express* [MS 3] which allows execution of an attachment in an email message simply by viewing the infected HTML message. This way Nimda activates immediately.

PDF Worm

Discovered in August 2001, Peachy worm is a proof of concept that a PDF application can be affected by a virus. In fact, Peachy is another VBS mass-mailer that uses the CreateObject function to open *Outlook*, but in this case the VBS file is embedded in a PDF file which is attached to the worm's email message. The worm activates when a user opens the PDF attachment. This makes Peachy another chain letter.

The VBS worm code searches for email addresses in *Outlook* folders and for the first three email addresses in the contact folder, but it does this in a very complicated way (not exactly as the Outlook.Application&AddressLists method does). However, it creates its message (CreateItem), builds the message subject and body, attaches the PDF file (Attachments.Add) and sends the message (Send) in a way similar to the other CreateObject methods.

Peachy affects only the users of the full version of *Adobe Acrobat*, and does not spread on systems on which only the common *Adobe Acrobat* reader is installed.

MSN Messenger Worms

Another application that was targeted during 2001 is *MSN Messenger*. Two new worms were developed for this application, Choke and Newpic. While the IRC chat clients are designed for many-to-many real-time communication, *MSN Messenger* is designed for one-to-one communication.

Choke and Newpic both use the same method of spreading via *MSN Messenger*. These are *Windows* executable files written in Visual Basic. Once the infected file is executed it creates a copy of itself on the victim's machine (choke.exe in the case of Choke). Then it adds a run key to the registry so that the worm will be executed when the system is restarted.

To spread further, the worm waits for an incoming message. Once an incoming message is received, the worm replies to it with its own message, making it appear as if the message has come from the person whose machine it has infected. In its message, the worm tries to encourage the other person to send a request for a file. Then, depending on the answer, it sends the worm's copy disguised as the requested file.

In order to propagate, these worms need to use social engineering, in order to persuade the recipient to request a file and subsequently click on it.

Gnutella Worms

An application that was attacked in May 2000 is the *Gnutella* peer-to-peer file sharing application, which is similar to *Napster*.

The polymorphic GWV worm, written in VBS language, was the first worm to attempt to affect *Gnutella*. When an infected VBS file is executed, it creates several copies of itself to the *Gnutella* installation directory 'C:\Program Files\gnutella' with various file names. To enable execution it modifies the gnutella.ini file in that directory by adding the .vbs extension to the list of extensions allowed. To spread, the worm adds the *Gnutella* installation directory to the list of the shared directories in gnutella.ini. This enables other *Gnutella* users to download and execute the worm [S. Rautiainen 2].

Another method of spreading via *Gnutella* has been used in the W32/Gnuman worm, also known as Mandragore or Gspot. Initially this worm connects to the *Gnutella* network as one node. Then it monitors the requests there and answers each of them by sending a copy of itself disguised by using the name of the request as its file name with a .exe extension. The worm spreads further when the next user receives the 'requested' infected file and clicks on it. The fact that Gnuman answers each request means that infected nodes become overloaded, thus making the infection obvious [M. Hypponen].

The Gnuman worm was quite widespread among *Gnutella* users at the end of February 2001. However, worms using this method of spreading require *Gnutella* to be installed on the user's computer and, since this is not a commonly-used application, they cannot be considered a big threat.

IIS Worms

IIS worms are worms that infect Web servers which use *Microsoft Internet Information Server* (*IIS*). They are also called Web worms. During 2001 there were two big cases of *IIS* worms: CodeRed and Nimda.

i) CodeRed – Fully Resident

CodeRed uses a security vulnerability known as Index Server Extension of *Microsoft IIS* [MS 1], or buffer-overflow. Using this vulnerability, CodeRed executes code on a remote Web server that runs *IIS*. The malicious code travels in HTTP request, so no actual file transfer is made.

When the worm activates on a server it searches for vulnerable hosts by scanning a massive number of random IP addresses and infects them if they are vulnerable. The random search for IP addresses was improved upon in subsequent versions of CodeRed so it searches for neighbour networks more aggressively [G. Erdelyi 2].

The fact that CodeRed travels without file transfer makes detection very difficult. In order to spread it does not need any human assistance. CodeRed is a typical example of an auto-worm.

ii) Nimda – Uploads and Activates

While CodeRed spreads from one infected server to another, Nimda infects workstations as well as servers, thus having a greater impact on end users.

Nimda spreads in four different ways: as file virus infector, as an Internet worm (email worm and Web worm) and as a network worm.

To infect Web servers Nimda uses an exploit known as Unicode exploit [MS 2]. Nimda searches the Internet for vulnerable *IIS* Web servers. Once a vulnerable server has been found Nimda uploads itself to the server using TFTP as protocol and activates that copy.

When activated on the server the worm modifies the Web page file contents by adding its code. The worm searches for Web server files with extensions .htm, .html and .asp, and with names 'index', 'main', 'default' and 'rename'. When it finds these it creates a file called 'Readme.eml' in the same folder. This is the email form of the worm – a multi-part MIME message with the worm code MIME-encoded inside.

The modification of the Web page files consists of adding a small piece of Java Script at the bottom of these files. When a user browses the infected Web site with a vulnerable *IE* browser [MS 1], the infected Web page file is opened and the Java Script executed. The worm downloads and executes on the remote machine automatically.

Network Worms

Network worms are worms written to spread through a local network. While Internet worms use several different methods of spreading, a network worm uses a simple method consisting of the following steps: it searches for shared drives; maps these drives; copies itself to these drives and executes its code.

The worm does not necessarily execute immediately. A network worm might execute by copying its code in the Start Up folder to run later when the machine is restarted.

VBS/Netlog is an example of a network worm. Using a combination of random numbers Netlog generates random IP addresses, then searches for a shared C: drive on the entire subnet space (from 1 to 254). When it finds a shared C: drive the worm maps it as J: drive. Next, Netlog copies its code using the copyfile function (copyfile c:\network.vbs) to the *Windows* and *Windows* Start Up folder of the shared remote drive so that its code will be executed when the machine is restarted.

The method this virus uses to search for IP addresses could result in finding not a local but a global IP address, meaning that VBS/Netlog could spread via the Internet. Many of the existing script email worms also contain a routine that enables them to spread through the local network.

Conclusions

While the CreateObject function and the AddressLists methods of spreading have been very popular in worms over the last two years, in 2001 the method that uses its own SMTP engine became more common. Worms that use this method attempt to avoid dependence on the email client.

Experience from this year suggests that we can expect to see more email worms, such as Nimda and Klez, that activate on reading an infected email message (active email worms). These worms try both to achieve fast spreading and to avoid detection by scanners.

All-in-one worms (i.e. combination-methods of spreading similar to those used by Nimda) will make the removal of worms very difficult. Virus-worms (such as Nimda and Klez) for which disinfection is not an easy task for many anti-virus products, will remain for a long time.

Since users in general are not very concerned about computer security, we might see more worms using vulnerabilities. Unfortunately there are a lot – Nimda itself uses 16 of them.

The scenarios of the 'Warhol Worm' [N.C.Weaver] and the 'Flash worm' [Stanford, Grim & Jonkman] – worms that could paralyse the Internet in minutes or even seconds – sound very worrying. However, experience shows that explosively spreading worms tend to have a short lifespan [V. Bontchev 2]. Even a very fast 'flash' worm cannot spread so fast – not in 30 seconds nor even in 15 minutes.

Whatever happens, the development of the Internet worms, as a fast way of spreading, will continue in the future.

References

[G. Erdelyi 1] Virus description database, *F-Secure Corporation:* http://www.F-Secure.com/.

[G. Erdelyi 2] Personal communication.

[I. Muttik] 'Time to Shiver', *Virus Bulletin*, October 1998, pp. 9–11.

[J. Shoch] 'The "Worm" Programs – Early Experience with a Distributed Communication', *Computer Practices 1982*, March, vol 25, pp. 172–180.

[K. Tocheva 1] 'From VBS to VBA', *Virus Bulletin,* March 1999, pp. 6–8.

[K. Tocheva 2] 'One Sharp Corner', *Virus Bulletin,* December 1999, pp. 8–9.

[K. Tocheva 3] 'Multiple Infections', *Virus Bulletin 1999 Conference Proceedings*, pp. 301–313.

[M. Hypponen] Virus description database, *F-Secure Corporation*: http://www.F-Secure.com/.

[MS 1] Index Server Extension of *Microsoft IIS*: http://www.microsoft.com/technet/security/bulletin/ms01-033.asp.

[MS 2] Unicode exploit, Unicode Directory Traversal vulnerability, http://www.microsoft.com/technet/security/bulletin/ms00-078.asp.

[MS 3] MIME Vulnerability, Incorrect MIME header can cause *IE* to execute email, http://www.microsoft.com/technet/securitybulletin/MS01-020.asp.

[N.C. Weaver] 'Warhol Worms: The Potential for Very Fast Internet Plagues': http://www.cs.berkeley.edu/~nweaver/warhol.html.

[P. Ször] Virus description database, *F-Secure Corporation*: http://www.F-Secure.com/.

[S. Rautiainen 1] 'Is *Linux* Security for Real?', *Virus Bulletin 2001 Conference Proceedings*, pp. 329–338.

[S. Rautiainen 2] Personal communication.

[Stanford, Grim & Jonkman] 'Flash worms: Thirty seconds to infect the Internet' http://www.silicondefense.com/flash/.

[V. Bontchev] 'Methodology of Computer Anti-Virus Research', Doctor thesis, University of Hamburg 1998, pp. 17–23.

[V. Bontchev 2] Personal communication.

FEATURE SERIES 2

Combating Viruses via Email Part 2

Carlos Ardanza
Panda Software, Spain

[In this two-part article, Carlos Ardanza describes the means offered to the anti-virus industry by the two biggest players in the mail applications market to combat viruses, in both clients and servers. In last month's instalment he looked at Microsoft, and in this, the concluding part, he considers the contributions of Lotus - Ed.]

Lotus produces the well-known email and groupware system *Lotus Note*s/*Domino*. Until recently, very few anti-virus manufacturers had developed protection for the client.

An anti-virus product for the *Lotus Notes* client has the same mechanisms for controlling the flow of documents as an anti-virus product for the server. For this reason, I shall focus on the server protection.

The only difference that does stand out is that the *Notes* client has a transfer optimization system to improve performance and reduce network traffic. This system is based on cacheing the operations that are carried out on the databases in the remote server. This means that the anti-virus product cannot open the original objects unless it opens the database again, resulting in loss of efficiency and a load on the network. This is certainly something that *Lotus* needs to improve.

Lotus Domino Servers

In addition to offering software developers an API to access the *Notes* documents databases, *Lotus* offers two events systems that anti-virus products use to protect *Notes* document databases.

Hook Driver

The Hook Driver system was implemented by *Lotus* in version 3.x and remains available in later versions for compatibility reasons.

The Hook Driver system consists of a number of events that inform the user every time an operation is carried out on a document. In order to use the system, it is necessary to develop a DLL and provide a pointer to a function that will be called every time an event occurs. This system is very limited as it offers only three events.

Extension Manager

The Extension Manager system was implemented by *Lotus* in version 4.x. The idea and philosophy behind the Extension Manager system is very similar to the Hook Driver system, however it has two significant advantages:

- It has more than 160 events (as opposed to three in the Hook Driver system) that allow an application to be controlled down to the finest detail of any operation carried out on a *Notes* database.
- It works with all *Notes* tasks. It is, without doubt, a vital feature for developing an effective anti-virus product, as it is important to protect the server task. However, it is equally important to protect others, such as the router task – which, after all, is where external infections will come from and is the point at which you can prevent an internal infection from reaching clients, suppliers or other companies with which you have exchanges via email.

Therefore, anti-virus products that do not use the Extension Manager system (which at the moment are the majority) need to find alternative ways of protecting the router or other server tasks.

Database Protection

First, it is important to mention that there are anti-virus products that protect documents when they are opened and modified; other anti-virus products intercept only one of these two operations and there are others still in which the interception of these operations is optional.

As you might imagine, the best solution is that which protects all possible operations, however it is also the solution that demands the highest performance from the anti-virus product for it to be viable.

Regardless of the hook system used, there are two techniques for protecting *Notes* databases.

INLINE Technique

This is the most secure technique, because it scans documents as they are opened or modified and the client cannot access them until they have been scanned and disinfected. The INLINE technique is the one that demands the highest performance from the anti-virus product.

QUEUE Technique

Every time an open/modify event arrives, the anti-virus product places the document in a scan queue and frees it up so that the document can be accessed immediately by the client.

The QUEUE technique demands lower performance from the anti-virus product, but it is a technique that does not

offer even the slightest security guarantees. In addition, this system can cause replica conflicts as, in its disinfection task, the anti-virus product modifies a document that the user has open. There are still some anti-virus products that use this technique.

In my opinion, the best *Notes* database protection is that which meets the following conditions:

- It scans when files are opened and modified.
- It uses the INLINE technique.
- It scans files without extracting them to disk.
- It uses a cache system that prevents a clean document from being scanned several times, unless it has been modified.

Router Protection

As the router cannot be protected with a hook driver, the anti-virus products that use this system have a second task which checks the router database MAIL.BOX continually. This technique is known as 'dead message scan'.

'Dead Message Scan' Technique

The anti-virus task checks the MAIL.BOX database, looking for new documents. When it finds a new document in this database, the document is marked as 'dead' so that the router does not route it. Next, the document is placed in a scan queue and once it has been scanned the 'dead' marker is removed. From then on, the router can handle the message and send it to the recipient. This technique has the following shortcomings:

- There are two tasks that are pooling the same database simultaneously, even though this system is designed so that only the router task accesses the database. The router polls once per second by default and the anti-virus product should do it a lot more frequently. Therefore, on the one hand this technique results in bottlenecks, and on the other does not guarantee that all documents will be scanned before they leave the router.
- In large- and medium-sized servers, the simultaneous blocking of the database by several tasks leads to blocks and the corruption of the MAIL.BOX database.
- Some documents can reach the next server marked as 'dead', which means that they will be waiting in the MAIL.BOX of this server indefinitely.
- Version 5 of *Lotus Domino* included the capacity for several databases to be available in the router. The anti-virus products that do not adapt to this new condition will protect only one of the databases used by the server for routing.

In spite of the shortcomings described above, the majority of anti-virus products use the 'dead message scan' technique.

Hook Driver vs. Extension Manager

In my opinion, the best technique for scanning the router is through the Extension Manager. Unlike Hook Driver, Extension Manager allows the integration of anti-virus products into the functionality of the router and it is the router that passes the control of the document to the anti-virus so that it can be scanned and disinfected (if necessary). After this the router takes control again so that the clean document can be routed.

In addition, the Extension Manager technique eliminates the need for the anti-virus product to consider the number of databases used by the router. In this case the router and the anti-virus product work together simultaneously and naturally. If this is combined with a scan that does not extract files to disk, the result is a very efficient anti-virus product.

Limitations

Some of the main limitations of the *Lotus Notes* APIs that we have come across while developing our anti-virus product are the following:

i) Extracting files to memory

As I mentioned in the first part of this feature, *Microsoft* offers programmers a very powerful API for working with attached files, without extracting them to disk. However, in the case of *Lotus* there is not a simple universal system for working directly with attached files.

The solution adopted by almost all anti-virus products is to extract the files to hard disk in order to scan them with more or less sophisticated variants. It was a challenge for us to overcome this pitfall, but after a long research process, we were able to remain true to our philosophy of scanning files in their natural environment. This functionality meant that we were able to scan every file six times faster than before.

ii) Renaming files

Incredible though it may seem, there is not a function that allows the file extension to be changed. In this case, the only solution is to extract the file to disk, rename the file on disk, delete the attachment of the document, insert the new document and save the changes made to the document.

Fortunately, this operation is optional and usually it is performed only on infected files that cannot be disinfected – therefore it does not involve a very high load on the server.

iii) Deleting files

A simple method for deleting files attached to documents does not exist, as although the file is deleted, the icon that represents the document still exists.

Conclusions

We can conclude by saying that the two Groupware system developers have followed different paths.

Microsoft set off with the handicap of not having a good hook system for anti-virus products to be able to intercept server operations with all of the necessary guarantees, although it had a very powerful and flexible API for handling messages (MAPI).

Microsoft's next step was to design a specific API for anti-virus products (AVAPI), but without involving anti-virus developers in its design. The result was disappointing to say the least. However, *Microsoft* knew how to react and must be congratulated for the design, stability and performance of the new VSAPI included in *Exchange 2000 Service Pack 1*.

From version 3.x *Lotus* included the Hook Driver system of events. Although its functionality was very limited as it offered only three events, this system had the advantage that the events were synchronous.

In version 4.x, the system improved significantly with the introduction of *Extension Manager*. This includes all the events that an anti-virus product manufacturer could dream of and, more importantly, these events work in all *Notes* tasks. In this aspect, *Lotus* has an advantage over *Microsoft*. Now *Lotus* must address the limitations of its API when handling files. In particular, the difficulties involved in accessing them without extracting them to disk.

As a final comment, I would mention that, of the two manufacturers, we found *Microsoft* to be much more open to offering support, help and collaboration in the development of our anti-virus products.

PRODUCT REVIEW

RAV AntiVirus for Sendmail & RAV AntiVirus Desktop for Linux – Part 1

Matt Ham

This is *VB*'s first review of a *Linux*-based product and has expanded, as a result, into a two-part review. The growing popularity of *Linux* as a replacement for more expensive operating systems (as most of them are) has resulted, for both direct and indirect reasons, in a larger number of products becoming available for the platform.

The most obvious reason is that an increase in the number of installations of *Linux* has meant a greater demand for software on that platform – but, in this case, this is perhaps the less important part of the equation.

Indirectly, the popularity of *Linux* has brought about development activities of both the helpful and the not so desirable type. In the former camp is the Milter feature for *Sendmail*. This is a mail filtering interface which both simplifies the development of anti-virus software and enables the software to be potentially much more efficient for the same degree of effort.

In the less-helpful camp come the *Linux* viruses – these are not common as yet, however they are certainly not the zero-threat that they once were. On a non-*Linux*-related but relevant tack is the current preponderance of mass-mailing worms and viruses in the WildList.

Taking all of these into account, we have a platform of increasing popularity, with an increasingly easy method of implementing scanning within emails. Often *Linux* is used as a mail-processing operating system for networks where *Windows* machines are the majority of clients – and mail is the primary source of infection for these machines. It is not surprising that *Linux* anti-virus has gone from a virtual zero presence to a standard offering over the last couple of years.

GeCAD's Products

Away from generalities and onto specifics, what does *GeCAD* offer in this field? A name change for a start – the product range has changed from the presumably too parochial *'Romanian AntiVirus'* to the more international *'Reliable AntiVirus'*. Frequent references are made to *'RAV AntiVirus'* in *GeCAD*'s literature (which, if expanded, makes less sense than might be hoped).

The two products examined on this occasion were the on-demand scanner for the *Linux* platform – standard *VB* fare – and the product for *Sendmail*, which posed a few new challenges. An email-scanning product demands infected emails and plenty of them; details of the testing process will be covered later.

Other products from *GeCAD* in the anti-virus realm are for the generic desktop *Windows* platforms, *Exchange Server*, *BeOS*, *FreeBSD*, *OpenBSD*, *Solaris 8* and *Unixware*. Of these, the various Unix-based platforms are available only as mail gateway options. The various mail applications supported include *Sendmail*, *Qmail*, *Postfix* and *Communigate Pro*. *GeCAD* is not a company that has an interest only in anti-virus matters, but also offers bespoke solutions in areas which fall outside the realm of this review.

The Package

The boxed version of the *RAV* product supplied for testing tended towards simplicity and frugality as opposed to the massive collections of ephemera favoured by some companies. The design of the box features *RAV*'s trademark pet cat and an orange-brown colour scheme, which may soon see a change, given that the *RAV* Web site and GUI are currently both undergoing cosmetic enhancements. The packaging suffered from a rather flimsy construction, resulting in the arrival of the boxed product in a state somewhat flatter than I imagine it had been when posted, though the contents were undamaged.

The contents of the box comprised a jewel-cased CD, licence agreement, registration card and an A5 manual for the gateway product. The licence agreement is of the standard type, in which *GeCAD* says, in effect, (as does virtually every other anti-virus company) 'we don't claim this software does anything good and if it does anything bad it is your problem, not ours.'

The registration card is slightly more interesting and contains the software activation code, proof of legality and licence coverages as well as a registration form for the software. Other than the CD this leaves the documentation as the only other item of interest.

Documentation

Before describing the documentation it should be noted that *GeCAD* claims that most of its software is distributed electronically, and thus considers the physical documentation is of less importance than the electronic versions. Even with this proviso the documentation is disappointing, consisting of some very general background material and the man pages for the software.

The documentation claims that setup procedures will be described, but this is one of the areas in which information is at its sketchiest. It is probably fair to assume that *Linux* users are, by and large, well versed in installing packages, the RPM format being not exactly overbearing in complexity, but the brevity of this part of the documentation is, perhaps, taken a little too far.

The general background information provided covers an overview of viruses written in good detail, which serves well as an introduction. There is, however, a tendency towards slightly odd translations, including the remarkable, 'One who thinks that the virus is a malefic and genial creation of a programmer is wrong,' which serves to add little but confusion to the proceedings. This is followed by a small advertising spiel for *RAV* and the array of acronyms associated with its various features.

The program information follows and, as mentioned previously, is limited to the man page descriptions for the various modules. These man pages omit some information which comes in useful during installation, which is covered further below.

The CD

The CD contents are not limited to the products on review; the whole *RAV AntiVirus* product range can be found on the CD in addition to documentation, licence information, update files and tools. These tools are limited at the present time to a pair of removal utilities for W32/SirCam and W32/Nimda.

Documentation is provided in a mixture of .DOC and .PDF formats, with the latter reserved mostly for the mail gateway components. This is rather unfortunate since the *Linux* version of *Acrobat Reader* is not provided on the CD, though the *Windows* version is provided.

Primarily the language supported in the documentation is English and, although a selection of German and Romanian texts are supplied, these are few in number. Unfortunately the same problems that were apparent with the printed documentation are repeated on the CD, in that the two sources of information are more or less identical in their brevity.

As far as support is concerned, one year of support is offered as standard, with additional yearly increments available for 20 percent of the original purchase price. Support as defined here is not just technical support but includes virus database updates and major program updates too. There is an international distribution network which provides support in addition to sales, ensuring that support can be obtained in a wide selection of languages.

With licences in mind, this is a good point at which to discuss exactly how the various licence options for *RAV* work and to give a more detailed description of which products are available. Like most *Linux* applications there is a reduced price for non-commercial use, here a nominal $1.99 for *RAV Desktop for Linux Home-Users*. The prices referred to, and the review comments, relate to commercial versions of the *Linux* products, which have several subdivisions.

The statically linked version of *RAV Desktop for Linux* was not used, instead concentrating on the library reliant version which can be considered as the more standard version –

these have identical licences. The *Sendmail* offering has two versions from a technical viewpoint – one using LibMilter, the other being declared to be less efficient but having an identical licence status to its close kin once again. The licence differences for the *Sendmail* product are accounted for by there being different pricing according to the number of domains that are to be protected by the software – the number of mailboxes within any single domain is not an issue.

Web Presence

The two URLs relevant to *GeCAD*'s *RAV* products are http://www.ravantivirus.com/ and http://www.gecad.ro/, with the latter being dedicated to the whole gamut of the company's activities.

The home page of http://www.ravantivirus.com/ is crammed with information. The Web site is much less hierarchical than many, which explains the frenzy of information on the home page. The advantage of such a design is that, in most cases, there is not far to navigate in tracking down an area of interest.

From the home page a number of news items and virus alerts can be reached – these account for the bulk of the page. Smaller links take the user to product descriptions, electronic product orders, updates and downloads, support, feedback, online scanning, general virus information and technical support. As if this were not a large enough selection, *GeCAD* partners, product awards, mailing and discussion lists, a virus library, virus detection statistics and a collection of various other links can be reached from this page also.

Of particular note from *Virus Bulletin*'s point of view was the option on the *RAV* Web site to perform a local scan for viruses. Tempting though it might be to run the *VB* test set in this manner, all our viruses are well and truly locked up far away from Internet-connected machines and so a definitive test of this functionality remains outside the scope of the ethical. On a more practical note, the online scan, which uses the *RAV AntiVirus for Linux* engine, can scan only single files rather than directories or drives which makes it less useful than it might appear.

As this review was in its final stages, the W32/Badtrans.B outbreak was gathering momentum. This provided a good opportunity to gain some idea of the speed of reaction and capacity of the *RAV* Web site to handle such an outbreak. In a reassuring manner, details were up on the site speedily as were updates allowing for the detection and (admittedly not complex) disinfection of this worm. Also gratifying to see was that the download speed of the Web site did not appear to suffer any appreciable degradation when tested on the day – in fact, it seemed rather more speedy than on several previous occasions.

The most unusual content on the Web site is the area in which there are a number of suggested new designs for the *RAV Desktop* GUI. Visitors are encouraged to make their preferences known by voting for the 'cutest'.

As for the information presented on the Web site, this is updated regularly. A major layout change was implemented during the writing of this review which, though resulting in some odd but ignorable glitches with the graphics, made matters a great deal clearer than they have been in the past.

Installation

Installation of *RAV Desktop* from ravlin8.rpm package was a simple affair, though not without a couple of hiccups. When the package is activated under SuSE it spawns the YaST2 setup program which looks for dependencies and checks for possible problems during the setup procedure. In this case YaST2 declared that no problems were imminent, though during the installation process alerts were produced declaring that the group 'test_mailrelay' was not present and that the root account was being used instead. This problem could be ignored or the group in question could be supplied prior to installation, as there seems to be no obvious effect on operation. Whether this introduces security holes by unnecessarily setting parts of *RAV Desktop* with root access is a different matter, and the documentation makes no mention of this group.

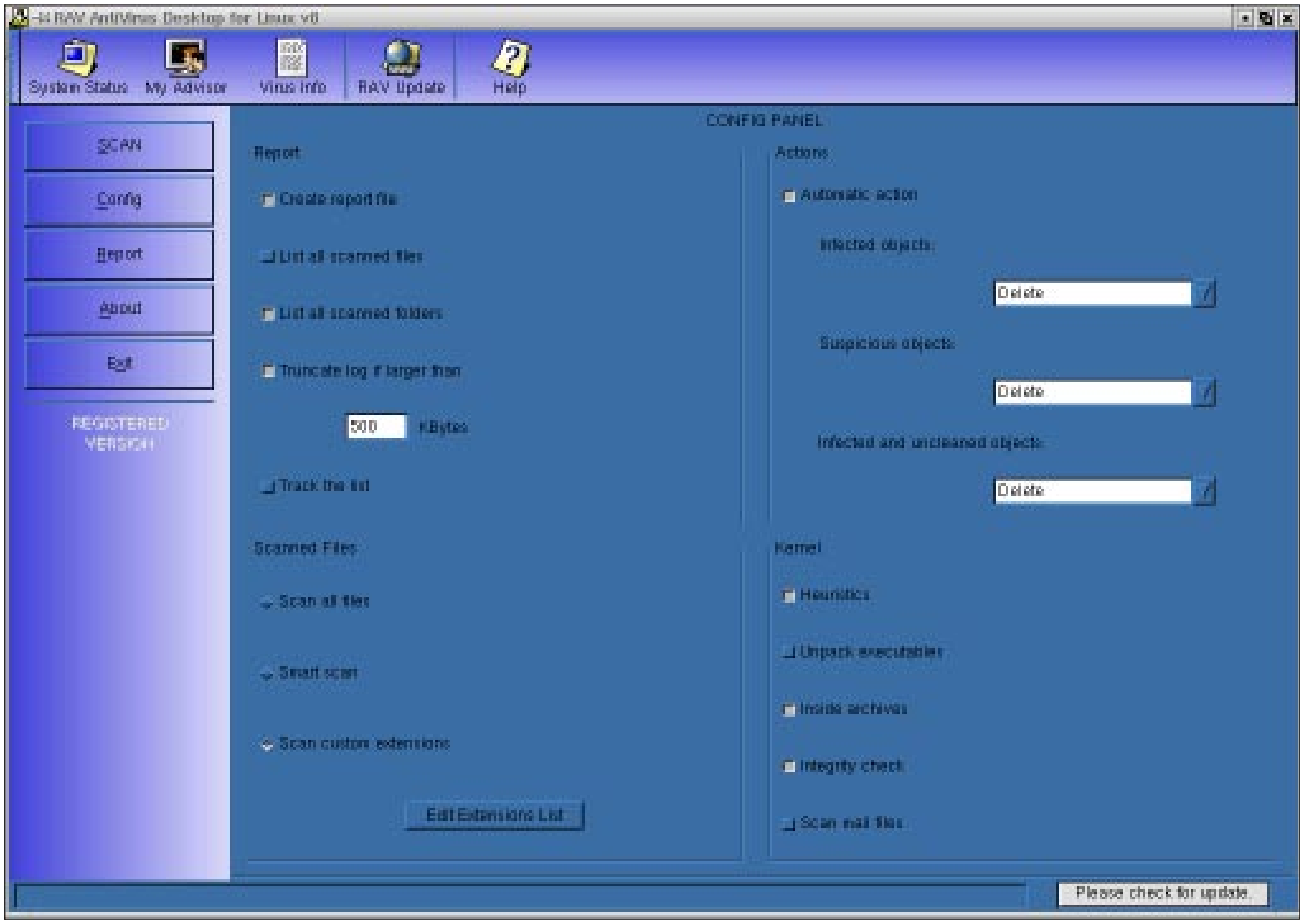
At this stage invoking ravlin8 from within KDE2 produced a familiar GUI. However, attempting to execute as a shell command caused a file-not-found and usage error. The correct usage was declared to be ravlin8 -h which was duly tried – and the same error message resulted. Both this and the previous oddity were reported to the *RAV* developers, who admitted that these were problems in the review version, but said that the problems have been removed from current versions of the program.

Installation of the *Sendmail* filter scanner was a considerably more involved affair. The documentation does not stress the fact that multiple packages are required to install the filter program itself – both the ravcore and ravmd packages must be installed, in order, before the *Sendmail* or ravmilter portion can be added.

If installed upon a machine which has ravlin8 installed already, or vice versa, this installation process requires that it writes over some common files between the two packages. This process reverts the version of the program to that of the last installed program, which is irritating when the upgrade and licensing procedure has just been applied to the original installation.

For *RAV for Libmilter*, the Libmilter functionality must next be compiled into *Sendmail*, which requires the user to have installed the *Sendmail* source package. This compilation process is not described in the documentation and the user is referred instead to the documentation for *Sendmail* itself – which is not altogether helpful.

Once this process has been completed, the *Sendmail* configuration file must be edited manually so as to invoke



Report
Create report file
List all scanned files
List all scanned folders
Truncate log if larger than 500 KBytes
Track the list

Scanned Files
Scan all files
Smart scan
Scan custom extensions

Actions
Automatic action
Infected objects: Delete
Suspicious objects: Delete
Infected and uncleaned objects: Delete

Kernel
Heuristics
Unpack executables
Inside archives
Integrity check
Scan mail files

the mail filter. Since sendmail.cf is legendary in its complexity, this in itself was alarming, though compiling in the Libmilter functionality (marked for future release and thus not officially supported), broke *Sendmail* completely, so this was a moot point. At this stage more manual tweaks of the *RAV* configuration files are required. I can only hope that in future releases at least some of this task is automated.

The installation of *RAV for Sendmail* was selected as being potentially a simpler method of testing the mail scan functionality, again, the greatest problem being lack of documentation. The ravmd portion of the product acts as a daemon ready to pass messages to the *RAV* engine for processing. Activating this was simplicity itself. The activities of the engine are controlled by a large, manually-edited configuration file, though thankfully the information within the base version of the configuration file is sufficiently detailed to make editing relatively simple.

Updates and Upgrades

Upgrades and updates of the products are in fact integrated as one process and may be performed either automatically or manually with update files coming in several guises. Updates are available as Daily, Weekly and Full versions – varying here simply in the time period covered by the application of the update. If updates are performed daily, for example, the small Daily update files can be used, while upon an installation from an older CD the Full update will be a much better option.

For an arbitrarily selected day the sizes of these files in the custom RUP file format were noted to be 2 KB, 100 KB and 1269 KB respectively. These sizes are specific to the RUP files which are designed for use in automatic update procedures. Plain archives of the update files are available too, which for the same day came in at 1 KB, 8 KB and 1262 KB respectively.

The figures show that packaging as the RUP file does add a slight overhead to Daily and Full update packages but, mysteriously, it expands the Weekly package by a factor of twelve, which looks odd in the extreme.

With all these files on offer how are they used? The RUP files are those used in automatic updates and if in the right place can be automated easily. The *Sendmail* scanner is, in this case, much more flexible than the *GUI Desktop* scanner, for the latter demands that updates are performed from the *GeCAD* servers while the former may be pointed at local machines.

Since the *VB* test network is very much isolated from the real world (a useful precaution when sending thousands of

Hard Disk Scan Rate	Executables			OLE Files			Zipped Executables		Zipped OLE Files		Virus Test Set	
	Time (s)	Throughput (MB/s)	FPs [susp]	Time(s)	Throughput (MB/s)	FPs [susp]	Time (s)	Throughput (MB/s)	Time(s)	Throughput (MB/s)	Time(s)	Throughput (MB/s)
Default settings	606	902.5	[1]	11	7212.2		236	675.5	19	3926.7	751	N/A
Smart extension selection	605	904.0	[1]	10	7933.4		220	724.6	18	4144.9	750	N/A
Custom extension list	606	902.5	[1]	N/A			N/A		N/A		618	N/A
Default with no heuristics	607	901.0		10	7933.4		233	478.7	19	3926.7	748	N/A
RAV 8 for Windows	612	893.7	21 [1]	42	1888.9		124	1285.6	52	1434.8	N/A	

infected mails), the Plain archived files were used, which are packaged in the ubiquitous ZIP format where more than one file is offered.

In cases where only one file alters, for example the addition of virus data as a VDM file, this is not packaged at all. In either case the contents of the package are simply copied over the appropriate files in the *RAV* installation directories – though consistency by packaging all files, regardless of their lone nature in some cases, might make automatic scripting of such updates simpler if required.

The Interfaces

The interface for *RAV Desktop* was the GUI, which is not quite identical but markedly similar to the interface on *GeCAD*'s *Windows* anti-virus products. The differences lie mainly in the complexity offered, since there is no need to configure on-access scanning and the feature set is, at first glance, slightly streamlined.

The interface has four main pages selected by large buttons on the left of the GUI which relate to 'SCAN', 'Config', 'Report' and 'About', with an extra button to exit the program. Additionally, in all views, a bar exists on the bottom right of the screen which notes how much time has elapsed since the last update, with a status summary of the program to the left of this.

Controls are offered in the form of large icons above this main area, labelled 'System Status', 'My Advisor', 'Virus Info', 'RAV Update' and 'Help'. These change the nature of the interface completely and, other than the first two in the list, are fairly self-explanatory.

'System Status' provides a 'tip of the day', which is the same as that produced when the program is initiated. The tips consist mainly of various URLs and minor configuration issues which, while not immediately apparent are useful nonetheless, so the subject matter here is well chosen. Below the tip of the day are statistics on the last software update, scan engine and details of the machine upon which ravlin8 is being run. 'My Advisor' is simply a link to the *RAV* homepage which saves an administrator a little typing.

'SCAN' is the simplest of the main pages, offering a device and directory structure in which targets for scanning can be selected in the usual tick-the-box fashion. A large button starts the scanning process at which point the displayed information changes to the Report page.

The 'Report' page is that which is seen most frequently during testing as *RAV* works its way through the clean and dirty test sets. Most of this page is taken up with a listing of the files scanned and the actions taken if these are found to be infected. Above this is a selection of statistics concerning the current scan, which includes 21 different pieces of information covering such obvious topics as files scanned and infections found, together with the rather less commonly-noted I/O errors and kb/s statistics.

The 'Config' page is the main source of user interaction in this GUI version. The page is divided into quarters, with each being more or less concerned with one particular facet of program control. The user can determine whether or not reports are created and, if so, whether all scanned files or folders should be included in the list. Also, the log file can be truncated if it exceeds a specified size and the display in the 'Report' page of the interface can be chosen to scroll while reports are produced. This latter option is off by default – a sensible choice, since excessive amounts of scrolling while scanning large numbers of directories has traditionally slowed down many products.

Also, scanned files may be selected, with the default being the increasingly standard 'all files'. Alternatively, smart scanning of files may be selected or the scanning of a custom extension list used.

Moving on, there is a selection of drop-down menus where the action taken upon infection can be selected. There are no great surprises on offer here. A selection of toggles cover other functionality. Heuristics may be set on or off, the default being on, as can the unpacking of executables (where the default is off). This is not to be confused with the unpacking of archives which is the next option and where the default setting is on. Integrity checking may be set to its original activated or alternatively deactivated state and finally there is the option to scan within mail files which is deactivated initially.

The 'About' page is the least important portion of the interface in terms of day-to-day use of the software. It gives contact details for *RAV*, repeats the last updated information and carries a copyright notice.

Tests – Desktop

The *RAV for Linux Desktop* scanner was tested with local files with a variety of settings, all tests being on-demand since on-access scanning is not provided. The default settings are with heuristics on, and all files checked; this was taken as a baseline.

Additional checks were performed using the intelligent file check and defined extension list and by disabling heuristics for a total of four sets of detection results. Speed tests were performed over the clean set, on the same hardware as that used for Comparative reviews, so broadly comparable with figures there, and also on the infected scans for a purely self-contained comparison of relative scan speeds.

It became apparent quickly that the defined extension list was far more speedy than other settings but that this was not a particular plus point for its use. Speed was gained since the only extensions scanned by default were .COM .ELF .EXE .HTM and .VBS. This cut out a large proportion of the *VB* test sets both on the infected and clean sets but, in its favour, it does scan extensionless files by default.

Even with the option to scan inside archives, .ZIP files are not included in this extension list. When the matter of the extension list is ignored the difference in scan speeds between the various tests was almost totally negligible, which would seem to indicate that whichever is likely to give the best detection rate should be chosen. This is slightly simplistic since the intelligent file typing would almost certainly be faster than the all-files option if many uninfectable files were present on a machine.

Since these tests were performed on the same test sets as are used in the Comparative reviews and upon the same hardware, it is valid to compare the times between this test and the latest comparative. For the default settings of *RAV Linux Desktop* as used here in comparison with the default settings of *RAV 8* on *Windows NT Server* the results on clean executables were similar enough that they could be considered identical, while the *Linux* product was noticeably faster on zipped files. Unzipped OLE files were the only area in which a slower performance was noted for the *Linux* scanner than its *Windows* counterpart. The difference between *Windows* and *Linux* performance does not come as a great surprise. It should also be noted that while the hardware and test sets used for these platform comparisons was constant, the version number of the software was not.

One notable plus-point in these figures exists, and that is in the matter of the false positives noted in this set of testing. Rather than the vast number seen on past outings, the final score here was down to zero – clearly a degree of tinkering of a very positive nature has gone on behind the scenes.

Last on the agenda is the testing of detection rates. The default settings of the engine, targeting all files, would seem to be a good choice. However, by default, executables are not unpacked. This caused misses in a number of files where detection would otherwise be achieved.

In the Standard set a number of older, packed, executable viruses were missed, none particularly remarkable or dangerous in any modern environment. More worrying, however, were the misses of VBS/San.A, VBS/San.B and VBS/Val.A in the Wild set. All of the files mentioned so far were detected if executable unpacking was chosen as an option.

This leaves those files which were undetected whether or not these settings are altered, and by far the majority were polymorphics: some Cryptor samples, all W32/Tuareg.B and W32/Zmist.D samples and two samples of Pathogen were the offenders here. This left a very small but very important remainder in the standard test set undetected – some, but not all, of the W32/Nimda.A files, which are by now widespread in the wild.

This detection rate is good, with some slight misgivings when the absolute defaults are used, and becomes much more satisfactory when the packed executables functionality is activated. The missing of W32/Nimda.A files is worrying, but is the only real fly in the ointment.

When heuristics were removed from the equation, which had been noted before to change little in speed on clean files, there was very little difference in detection rate in all but the Standard set. Here, again, older files were those missed rather than anything new, interesting or dangerous.

For intelligent-file-typing, no differences were noted as opposed to the use of the all-files option, leaving it as a matter of personal preference and dependent very much upon the environment scanned as to whether speed increases might be expected from the intelligent-file-typing option. The *VB* test sets for, example, consist entirely of executable or infectable material and would thus be scanned in their entirety whether all files or intelligently determined infectable files were selected. In such a case scanning all files will be more effective since the speed gains by using intelligent file checking are negated and the chance exists that infected files will not be scanned if in a newly dangerous file-type.

In the case of a scanner in a less artificial environment, data files could make up a significant percentage of scanned files if all files were scanned. In such a case an intelligent file checker would score highly in the speed stakes.

Issues such as these are of great importance when selecting configurations and one of the reasons that *Virus Bulleti*n is unwilling to declare features or products universally good or bad in reviews.

RAV Review Part 2

The mail and content scanning portion of the program has not been mentioned so far in the tests, and the scanning of *Linux* malware – ELF binaries, Shell and Perl scripts – is also noticeable by its absence. These, and the final conclusion, will be dealt with next month in the second part of this review.

SUBSCRIPTION RATES

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

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

Editorial enquiries, subscription enquiries, orders and payments:

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

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

US subscriptions only:

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

Tel (781) 2139066, Fax (781) 2139067

END NOTES AND NEWS

The Black Hat Windows 2000 Security Conference takes place 7–8 February, 2002 in New Orleans, LA, USA. The conference will focus specifically on the security issues created in the *Windows* environment. Training sessions covering seven topics will take place before the conference, on 5–6 February. The call for papers for the conference remains open until 15 December 2001. For more details see http://www.blackhat.com/.

The VI Ibero American Seminar on Security Information and Communications Technologies takes place in Havana, 18–24 February 2002. Topics covered will include anti-virus software, network security, Web security and network remote diagnostics. For more information contact José Bidot: email jbidot@seg.inf.cu.

Information Security World Asia 2002 will be held 16–18 April, 2002 in Singapore. The show will include a wide-ranging exhibition, discussions of the latest security issues and a number of interactive workshops. For further information about the show visit the Web site http://www.isec-worldwide.com/isec_asia2002/.

Infosecurity Europe 2002 will run from 23–25 April 2002 at London's Grand Hall, Olympia. Over 40 free seminar sessions will run over the three days, explaining some of the key security issues facing organizations today. For more details visit the Web site at http://www.infosec.co.uk/.

The Southwest CyberTerrorism Summit, to be held 4 May, 2002 in Dallas, TX, USA, will feature presentations from both hackers and industry security experts. Topics include Wireless Hacking, Cyber-attacks, Information Warfare, Privacy, Computer Viruses, Industrial Espionage and Identity Theft. For more information visit Web site http://www.DallasCon.com/.

Infosecurity.de 2002, the international specialist exhibition for IT security takes place 14–16 May 2002, in Düsseldorf. For the first time an accompanying Specialist Conference will run throughout the exhibition period. For more details about the exhibition and conference see http://www.infosecurity.de/.

Information Security World Australasia 2002 will be held 19-21 August 2002 in Sydney, Australia. For full conference and exhibition details see http://www.informationsecurityworld.com/.

Sophos MailMonitor for SMTP* is available now for *Windows NT/2000. *MailMonitor* scans all email prior to it reaching the SMTP server; virus-free email is passed on to the server, while infected email can be quarantined, deleted or disinfected. *Sophos* is one of five British companies shortlisted for the Company of the Year Award in this month's Real Business/CBI Growing Business Awards. For more details visit http://www.sophos.com/.

AV-Test.org has completed a comparative review of AV products for *Windows 98*, *ME*, *NT 4*, *2000*, *XP Home* and *XP Professional*. The results are available on the Web site http://www.av-test.org/.

Based on a six-month study of the security market, ***Softwin* has changed the name of its flagship product *AVX/AntiVirus eXpert*** in order to 'better correspond to its actual technologies'. *AVX* will henceforth be known as *BitDefender* and as well as anti-virus protection, will include a personal Firewall solution. See http://www.bitdefender.com/.

A Convention on Cybercrime drawn up by the Council of Europe has been signed by 30 States, including the four non-member States Japan, Canada, South Africa and the USA. According to the Council's Deputy Secretary General, the Convention is intended to give national legal systems 'ways of reacting together to crimes committed against or through computer networks.' See http://www.coe.int/.

The UK's police National Hi-Tech Crime Unit (NHTCU) has appointed an industry liaison officer to develop a confidential crime reporting system by April 2002. The liaison officer has the task of creating a system to help companies report digital security breaches to the police without suffering embarrassing public disclosures. Earlier this year AnnaKournikova author Jan de Wit received what was widely considered an inappropriately light sentence for his role in the writing and distribution of the virus. This was believed to be due to a shortage of evidence in the investigation.

The staff of *Virus Bulletin* would like to wish all our readers a very
Merry Christmas & Happy New Year