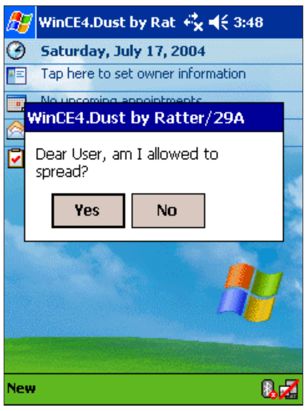
SEPTEMBER 2004

virus BULLETIN

The International Publication on Computer Virus Prevention, Recognition and Removal

CONTENTS

IN THIS ISSUE


WinCE4.Dust by Rat 3:48
Saturday, July 17, 2004
Tap here to set owner information
WinCE4.Dust by Ratter/29A
Dear User, am I allowed to spread?

VIRUS IN YOUR POCKET(PC)

Hot on the heels of the discovery of SymbOS/Cabir, the first virus to spread via mobile phone, comes the discovery of WinCE/Duts.A, a *Windows CE* file infecting virus. Eric Chien has all the details of the virus that politely asks, “Am I allowed to spread?”

page 4

AV TESTING IN THE SPOTLIGHT

What exactly do AV certification programmes test and how? Ferenc Leitold introduces the *CheckVir* anti-virus testing service and certification programme, while Matt Ham provides an overview of *Virus Bulletin*’s own VB 100% testing procedures and lays a few common misconceptions to rest.

pages 10 and 12

vbSpam supplement

This month: anti-spam news & events; an introduction to the Institute for Spam and Internet Public Policy (ISIPP); ASRG summary.

ISSN 0956-9979

virus BULLETIN COMMENT

"If the AV industry were getting started today, we would not choose the approach that we currently pursue."

Paul Gartside, McAfee Inc.

A CHANGE IN APPROACH

For the last 10 years I have worked in the AV industry, testing software to ensure that products are the best they possibly can be, within the constraints of commercial pressure and deadlines. During this time I have formulated some opinions about the AV industry and about how we approach a couple of key issues that affect our customers.

One such issue is the delivery of AV definition files. I read Rob Rosenberger's comment in the June 2004 issue of *VB* (see *VB*, June 2004, p.2) with interest. Rob makes some very salient points about virus definition updates – in short, he makes the point that updates are addictive and the situation is likely to become worse. He is right, *unless* we in the AV industry change how we view some things.

Part of the problem is the reliance by AV vendors on positive identification to determine risk and performance. The argument for positive identification is that it is required for safe virus removal. I agree that if you are going to write to a hard disk then you had better be doing it for a specific and known task. However, I think that waiting for infection to occur is leaving things too late in the cycle.

Editor: Helen Martin
Technical Consultant: Matt Ham
Technical Editor: Morton Swimmer
Consulting Editors:
Nick FitzGerald, *Independent consultant, NZ*
Ian Whalley, *IBM Research, USA*
Richard Ford, *Florida Institute of Technology, USA*
Edward Wilding, *Data Genetics, UK*

Consider the anti-spam industry. The industry really got going while the spam problem was (and is) at its height. The sheer volume of spam is such that positive identification is simply not viable. Anti-spam products use a combination of generic rules and a scoring system to determine whether something is spam or not. This may not be the best method – false positives and misses are higher for spam detection than they are for AV detection – but it is necessary to take a generic, heuristic or behavioural approach due to the massive scale of the problem from the get go. I firmly believe that if the AV industry were getting started today, we would not choose the approach that we currently pursue.

The AV industry has had 17+ years to build a model for detection and cleaning. The same model has been in use since there was just one virus to deal with. Today there are over 90,000. I think the fact that people even care about a specific number is a problem. How many unique spam messages are there? Do you care? It is becoming clear that for the longer term this will not scale.

The pot of gold at the end of the AV rainbow is zero-day detection: to be able to detect and *prevent* an item of malware or other undesired attack (rather than remove it post infection). In order to achieve this, reactive action (such as driver-writing) will have to become a thing of the past, making way for generic, heuristic and behaviour-based blocking. This means that if we are to measure our success, it needs to be on different terms to those we use today, and the AV industry's professional organisations and publications (such as *Virus Bulletin*) need to be at the front of the charge by determining what the proper measure of success in AV should be.

There should be two sets of tools; the primary approach consisting of the behavioural and generic products that prevent the attack from occurring in the first place. Where possible this should be done on the wire and at the perimeter – however, tiered AV/security applications down to the desktop/mobile device will become increasingly important due to the porous networking nature of the modern business environment. For those unfortunate enough not to have these in place when they are infected, the more traditional AV products will be used – but specifically in the context of cleanup only.

The question I will leave you with is: "Do I care that I had XYZ virus, or do I simply care that something bad was stopped and that it did not have to be written to my hard disk and positively identified before being dealt with?" The *real* definition of success is knowing that you know you prevented something bad from happening, you did it at the right time, and the user did not have to become involved.

NEWS

SECURITY OF HASH FUNCTIONS CALLED INTO QUESTION

The encryption field was thrown into a frenzy at the end of last month when the security of hash functions MD5, SHA-0 and SHA-1 was called into question. First, a collision in SHA-0 was uncovered by Antoine Joux; then a group of Chinese researchers released a paper which outlined methods of finding collisions in the MD4, MD5, HAVEL-128 and RIPEMD algorithms; finally, researcher Eli Biham of the Israel Institute of Technology reported at the Crypto 2004 conference preliminary research findings that indicate the presence of vulnerabilities in SHA-1.

In principle it is not possible to design a hashing algorithm that prevents the production of duplicate fingerprints (hash collisions), but the hashing algorithms are designed to make it very difficult to generate duplicate hash codes. It seems that, for MD5 at least, it is easier to do so than originally hoped. While there currently does not seem to be an easy way of faking an arbitrary hash code – thus limiting the usefulness of an attack – it does call into question the usefulness of these hashes as digital signatures.

A similar situation is true of SHA-0, but the evidence that the more widely used SHA-1 is likewise broken is not currently conclusive. However, the possibility that SHA-1 may be flawed is a cause for concern, since SHA-1 has become a legal standard for document signing – it is currently embedded in PGP and SSL and is the only signing algorithm approved for use in the US Government's Digital Signature Standard.

[*Next month's issue of VB will contain a more detailed look at the security flaws in these hashing algorithms and the implications for the anti-virus industry.*]

XP SP2 WREAKS HAVOC WITH AV

According to *Microsoft*, the newly released (and much anticipated) *Windows XP Service Pack 2* (*SP2*) "will establish stronger security settings that help defend against viruses, hackers and worms." However, the company warns that SP2 could cause loss of functionality in a number of programs as, by default, the Windows Firewall will block unsolicited connections to a user's computer. Unfortunately the programs said to experience "loss of functionality" (despite having had the many months of *SP2*'s beta period to prepare) include several anti-virus and firewall programs, including versions of: *CA*'s *eTrust Armor*, *Command AntiVirus*, *BitDefender*, *Kaspersky AntiVirus*, *Eset*'s *NOD32*, *ISS*'s *BlackICE*, *Norman Personal Firewall*, *Norton Antivirus 2003*, *Sourcenext*'s *VirusSecurity* and *ZoneLabs*' *Zone Alarm*. A full list of the programs affected can be found at http://support.microsoft.com/.

Prevalence Table – July 2004

Virus	Type	Incidents	Reports
Win32/Netsky	File	210,441	76.91%
Win32/Bagle	File	54,369	19.87%
Win32/Zafi	File	4,005	1.46%
Win32/Dumaru	File	1,293	0.47%
Win32/Klez	File	391	0.14%
Win32/Lovgate	File	326	0.12%
Win32/Mimail	File	300	0.11%
Win32/Mydoom	File	272	0.10%
Win32/Bugbear	File	257	0.09%
Win32/Swen	File	195	0.07%
Win32/Sobig	File	190	0.07%
Win32/Funlove	File	188	0.07%
Ebscan	Script	149	0.05%
Win32/Fizzer	File	125	0.05%
Redlof	Script	108	0.04%
Win95/Spaces	File	104	0.04%
Win32/Valla	File	102	0.04%
Win32/MyWife	File	82	0.03%
Win32/Parite	File	61	0.02%
Win32/Magistr	File	50	0.02%
Win32/Hybris	File	49	0.02%
Win32/Yaha	File	39	0.01%
Win32/Korgo	File	34	0.01%
Win32/Gaobot	File	33	0.01%
Laroux	Macro	29	0.01%
Win32/Evaman	File	25	0.01%
Win32/Gibe	File	25	0.01%
Win32/Lovsan	File	22	0.01%
Win32/Sober	File	22	0.01%
WYX	Boot	22	0.01%
Win32/BadTrans	File	21	0.01%
Win32/Mota	File	20	0.01%
Others[1]		254	0.09%
Total		273,603	100%

[1]The Prevalence Table includes a total of 254 reports across 59 further viruses. Readers are reminded that a complete listing is posted at http://www.virusbtn.com/Prevalence/.

VIRUS ANALYSIS

WINDOWS CE COLLECTS DUTS

Eric Chien
Symantec Corporation

Just one month after the discovery of SymbOS/Cabir (see *VB*, August 2004, p.4) comes the discovery of WinCE/Duts.A, a *Windows CE* file-infecting virus written by a well-known Czech virus writer nicknamed Ratter. On 17 July 2004 Ratter, who is a member of the virus-writing group 29A, appears to have sent his creation to a variety of security companies. He included the original dropper file, along with two infected *Windows CE* executables.

Windows CE executable files use the familiar PE file format. In addition, many of the high-level language APIs that are found on traditional *Windows* operating systems exist in *Windows CE* environments. Thus, many of the techniques and ideas developed for other *Windows* operating system families including entry point obscuring and cavity infectors also apply to *Windows CE* environments. Other architectural aspects of *Windows CE* are also the same or analogous to traditional 32-bit *Windows* operating systems including the use of DLLs, the process and thread model, window message queues, and the memory model where each process has its own virtual address space.

WINCE/DUTS.A

WinCE/Duts.A is the first *Windows CE* virus. It is a classic parasitic file infector, which appends itself to the end of the host file. WinCE/Duts.A is 1520 bytes in size and infects executable files on *Windows CE 3.0*, *Windows CE 4.0*, *Pocket PC 2000*, *Pocket PC 2002*, and *Pocket PC 2003*. The virus could probably execute under *Windows CE 2.0* (or previous), but most devices running *Windows CE 2.0* are MIPS or SH3 processors. The virus is written in ARM assembly and thus will execute only on ARM platforms.

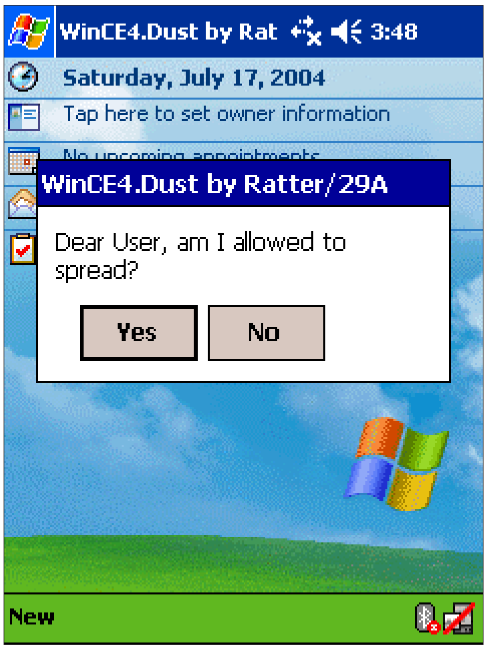
When an infected file is executed, the virus attempts to find coredll.dll, which is analogous to kernel32.dll, in the memory space of the current process. The virus finds coredll.dll by using a magic address that provides a pointer to a link list containing information about every loaded DLL. Each node of the linked list holds a variety of fields including a pointer to the filename of a loaded DLL, the base address of the DLL, and the offset to the export section.

WinCE/Duts.A iterates through the linked list comparing the DLL filename to 'coredll.dll' (in UCS-2). Once the node corresponding to coredll.dll is found, the virus obtains the base address and the offset to the export section. Using these values, the virus is able to parse the export section of coredll.dll, looking for particular exports by ordinal. The function pointers to these exports are stored on the stack and include:

__rt_udiv	MapViewOfFile
malloc	UnmapViewOfFile
free	FindFirstFileW
CreateFileForMappingW	FindNextFileW
CloseHandle	FindClose
CreateFileMappingW	MessageBoxW

This list of APIs is similar to other 32-bit *Windows* viruses with the exception of __rt_udiv which performs unsigned integer division and is used because ARM does not have a native division instruction like idiv. WinCE/Duts needs division for determining the padding size for file and section alignment.

Once WinCE/Duts.A loads the needed APIs, it politely asks the infected user if it can continue by displaying the following message box:


WinCE4.Dust by Rat 3:48
Saturday, July 17, 2004
Tap here to set owner information
No upcoming appointments
WinCE4.Dust by Ratter/29A
Dear User, am I allowed to spread?

If the user selects No, the virus returns to the host program. Otherwise, the virus continues.

The virus searches for files to infect by calling FindFirstFileW and FindNextFileW to find files matching the pattern *.exe. By not specifying an absolute pathname, the root directory of the device is searched (not the current directory as in other *Windows* implementations). The virus skips any files that are 4096 bytes in size or smaller and does not search any sub-folders.

Before infecting the file, WinCE/Duts.A verifies the file's characteristics. In particular, the virus ensures the file starts with 'MZ' and the PE signature is found. In addition, the file must have a machine type of ARM and the subsystem set to 0x0009, which is 'Windows CE GUI'.

Finally, it checks if the required operating system version field is set to 4.0. Interestingly, while Ratter may have believed that, due to this last check, his virus only infected *Windows CE 4.0* files, version numbers seen on marketing materials are actually stored in the subsystem version field of the optional header. For example, a *Windows CE 3.0* file will contain the operating system version number 4.0 and the subsystem version number 3.0. Thus, the virus will happily infect non-*Windows CE 4.0* files.

Next, the virus looks for its infection marker. The virus uses the 'Win32 Version' field in the optional header. If the field contains 0x72617461 ('rata') or if any of the previous checks fail, the virus simply skips the file.

Once a suitable file is found, the virus calculates its own size and adds it to the target file's size. If necessary this value is increased based on file alignment in the PE header of the target file. The virus calls CreateFileMapping on the target file with the new size value and thus automatically allocates space for itself in the target file.

The new entry point is set in the optional header to the end of the last section of the target file plus one, which is where the virus will copy itself. The distance between the new entry point and the old entry point is also stored within the virus body, allowing the virus to return to the host after completing its infection routine.

Once the new entry point is set, the virus expands the last section by modifying the raw data size and virtual size and adds the section characteristics, code, execute and read. The image size in the optional header is also updated to reflect the increased size.

Finally, the infection marker 0x72617461 ('rata') is set in the 'Win32 Version' field of the optional header and the virus copies itself into the expanded last section.

The newly infected file is closed and the virus continues to infect other files matching the *.exe pattern in the root directory until all possible files are infected.

After completing the infection routine, WinCE/Duts.A sets the PC (program counter, similar to the IP in x86 assembly) to the original host's entry point and thus returns control to the original host.

DUST THEORY

While WinCE/Duts.A's method of infection is rather straightforward, the virus does contain some more interesting internal strings which are never displayed to an infected user. The first string states:

```
This code arose from the dust of Permutation City
```

This sentence is in reference to the science fiction book *Permutation City,* written by Greg Egan who is not only an author, but also a computer programmer. In the book, Egan writes about simulations of intelligence including aspects of self-replication. In particular, he floats a theory regarding dust.

Egan's book is set in the year 2045 when physical simulations can occur, including making copies of people. While the copies run in a virtual reality, the virtual reality can interact with the real world. The main character attempts to create a world that allows immortality via his 'dust theory', believing that the universe is scattered as dust in random patterns, yet it forms what is seen as the universe. He uses this cloud of dust to create his own world, where the capital city is Permutation City.

Fortunately, by 2045, most readers who are anti-virus researchers today will have retired and out-of-control self-replicating simulated people will be someone else's battle.

AV RESEARCHERS

The other internal string states:

```
This is proof of concept code. Also, i wanted to make
avers happy.The situation when Pocket PC antiviruses
detect only EICAR file had to end ...
```

A lifetime of work exists already with the current *Windows* threats and there are no signs that the number of threats is decreasing. So I think that anti-virus researchers would be happier if widespread *Pocket PC* viruses never came to fruition.

SCIENCE FICTION TO SCIENCE FACT

While WinCE/Duts.A is the first public proof of concept of a *Windows CE* malicious threat, the ability to create viruses on *Windows CE* has been well-documented.

Furthermore, file-infecting viruses are the least worrying threat to the platform. Bluetooth, MMS (multimedia messaging), and other network protocol worms will be far more troublesome if Smartphones and PDAs ever become ubiquitous computing devices.

Fortunately, with the incompatibility of telephone networks and the diversity of handheld operating systems, worldwide impact on the scale of CodeRed or Netsky from a handheld threat is still only science fiction today – but that is likely to change in the next three to four years.

TECHNICAL FEATURE

ADVANCED VIRUS CRYPTANALYSIS

Mircea Ciubotariu
BitDefender, Romania

Successful cryptanalysis relies on encryption weaknesses; presumptions are at the heart of the technique. This article describes advanced virus cryptanalysis techniques and demonstrates how many troublesome polymorphic malware threats (such as the recent Win32.Bagle.{M-Q,S}@mm file infectors, aka W32/Beagle.{M-R}@mm) may be caught by a single cryptanalysis signature.

INTRODUCTION

In a previous article about cryptanalysis (see *VB*, November 2003, p.6) we discussed the basics of virus cryptanalysis based on the fact that most viruses use simple encryption techniques. These techniques reside mainly in logical/arithmetical functions found in the CPU's instruction set. Starting from their properties several weaknesses were deduced, which led eventually to the successful determination of encryption functions and their corresponding encryption keys.

Following the convention of our previous article we shall consider our encrypted block of data as a stream of units. A unit is defined as the amount of data – bit, byte, word, etc. – upon which the encryption/decryption function operates at any one time.

The process of encryption may be regarded as a transformation of the original data into a new meaningless bulk of data. To give back its meaning the encrypted data needs to be reverse-transformed. We have learned from the principles described that we can further transform the encrypted block – at the expense of some data – into a new form that is *independent* of the previous transformation.

Next we shall extend the approaches used for the most common malware encryption techniques, based on experience we have gained from direct contact with them.

NEG FUNCTIONS

The NEG function returns the negative value of its operand, so, for any A, NEG(A) = -A .

The NEG function operates like XOR(A, -1), except that it performs an additional increment, ADD(A, 1). The same can be said for a more general combination of the two functions, namely: XOR(A, -1) followed by ADD(A, K), where K may be any value. In XOR functions, we take -1 as being the value with all bits set to 1, since XOR does not consider signs.

Note that XOR(A, -1) is, in fact, NOT(A), the function that returns the complementary value of A based on the complementary values of the bits composing A. Also note that the order of the two functions in NEG is not important since NEG(A) is NOT(A) followed by ADD(A, 1) and is the same as ADD(A, -1) followed by NOT(A).

For a better understanding let's see how it works on some specific binary values, units being bytes:

```
A        01011101 (93)            00000000 (0)
NOT(A)   10100010 (162)           11111111 (255)
NEG(A)   10100011 (163 = -93)     00000000 (0 = -0)

A        00000001 (1)             11110000 (240)
NOT(A)   11111110 (254)           00001111 (15)
NEG(A)   11111111 (255 = -1)      00010000 (16 = -240)
```

One may say that 163 is *not* equal to -93, which is true, but when these numbers are represented in bytes (char and unsigned char types in C) they are the same because of the internal representation of integers.

The previous article showed how data encrypted by the ADD(A_i, K_0), ADD(K_0, K_1), …, ADD(K_{m-1}, K_m) function could be transformed to a sequence of (n - m) units, when n > m, dependent only on A_0, …, A_i, …, A_n or, put another way, independent of all K; K_0, …, K_{m-1} being the variable keys and K_m the constant key; A_0, …, A_i, …, A_n being the information currently encrypted and i being the current step in encryption.

This approach extends the polynomial reduction used for the ADD function with an additional NOT(A_i) using the following property of NOT applied on ADD:

given A_i' = NOT(ADD(A_i, K)) and A_{i+1}' = NOT(ADD(A_{i+1}, K))

then SUB(A_i', A_{i+1}') = NEG(SUB(A_i, A_{i+1}))

The trick is that the carry bit used for transportation in the ADD function works the inverse way on the NOT value:

```
NOT(ADD(A, K)) = ADD(NOT(A), NEG(K))
```

Furthermore, ADD is a commutative function; ADD(A, K) = ADD(K, A), so

```
NOT(ADD(A, K)) = ADD(NOT(A), NEG(K)) = ADD(NOT(K),
NEG(A)) = ADD(ADD(NOT(A), NOT(K)), 1)
```

After the polynomial reduction on such encrypted data we obtain the NEG value as if it was encrypted without the NOT scrambling and then reduced the same way.

Let's look at that with a practical example. We choose our units to be the following sequence of four bytes: 93, 0, 1 and 240. The sequence of the consecutive differences is 93, -1, -239, which is the same as 93, 255 and 17. We encrypt it by adding a constant value chosen at random, say 44, and we get 137, 44, 45 and 28. Computing the consecutive differences again we will get the same as above.

Now let's perform a NEG function on our original block of data. This results in -93, 0, -1 and -240 which, as unsigned values, are 163, 0, 255 and 16. Then compute the differences again: 163, -255 and 239. These are exactly -93, 1, and 239 – the previous values, but with changed sign.

ROTATION FUNCTIONS

ROL and ROR functions are uncommon functions for high-level languages like Java, C++ or Delphi. They are assembly instructions.

The chain of bits in a value is considered circular and ROL and ROR rotate it to the left and right respectively through the specified number of positions. They are complementary functions, which means that ROL(A, K) is the same as ROR(A, -K).

The big question is: how could someone extract information independent of the rotation key from a rotated unit? The only things that remain constant after rotation are the links between any two consecutive bits. And that is useful only for circling around indefinitely.

A reasonable solution was suggested by a colleague, Alex Carp: the number of bits with a chosen value, 0 or 1. Let's take for example the number 193 represented in bytes (11000001 binary). Rotated in any position it will always have three bits of 1 and five of 0.

When counting the bits of 1, the value 0 (zero) is a special value because it has no bits of 1. Correspondingly, when counting zero bits the value 255 is a special one because it has no bits of zero.

There are nine different ways to represent the number of bits of 1 – for example, for all the 256 values of a byte:

1. The value 0 which has no bits of 1.
2. The bytes with one bit of 1, eight values (1, 2, 4, 8, 16, 32, 64, 128).
3. The bytes with two bits of 1, 28 values (3, 5, 6, 10, …).
4. The bytes with three bits of 1, 56 values.
5. The bytes with four bits of 1, 70 values.
6. The bytes with five bits of 1, 56 values.
7. The bytes with six bits of 1, 28 values (252, 250, 249, 245, …).
8. The bytes with seven bits of 1, eight values (254, 253, 251, 247, 239, 223, 191, 127).
9. The value 255, which has all eight bits of 1.

The number of these possibilities may be computed using the following formula: $\log_2(n)+1$, where n is the total number of values (256 for bytes); $\log_2(n)$ represents the number of bits necessary to encode the n values – for a byte this value is 8 – and 1 comes from the special case mentioned above.

The solution in this case is to replace rotated units by their corresponding numbers of chosen bits. This gives $\log_2(n)+1$ numbers which contain $\log_2(\log_2(n)+1)$ bits of information; $\log_2((\log_2(n)+1))$ represents the number of bits necessary to encode the possibilities enumerated above, for a byte $\log_2((\log_2(256)+1)) = \log_2(9)$, which is approximately 3.17 bits.

Using this technique on a group of six bytes we obtain about 19 (6 x 3.17) bits which are independent of the random rotation of any byte from the group.

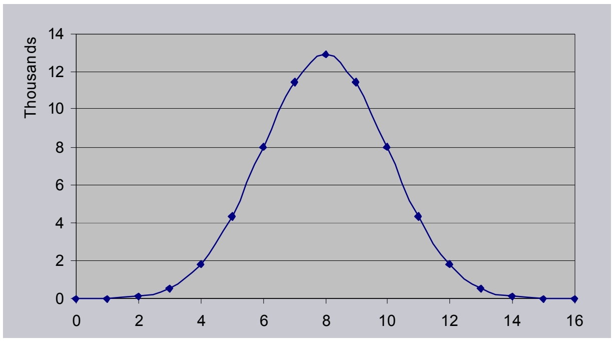
The difference between the total bits used for encoding $\log_2(n)$ and the number of bits used for this purpose $\log_2((\log_2(n)+1))$ is ignored because of the replacement, although there are techniques to decrease this loss slightly.

Moreover, the greater the value of n, the greater the information loss. For n = 65536 only about four bits, corresponding to 17 values, are used and almost 12 are unused.

There is another problem with this solution however: the distribution of units over the $\log_2(n)+1$ values. Most of them will be concentrated around the $(\log_2(n)+1)/2$ value and, for the first and the last units, will correspond to only one value each.

This leads to a new big question: how much information can be extracted independent of the rotation key? The correct answer to this and the appropriate approach for the rotation functions lies in the definition of rotation.

Rotations may be performed with a maximum key of $\log_2(n)$ - 1. If a unit is rotated above the number of bits it contains it over-rotates after passing the original configuration one or more times. For a rotation with $\log_2(n)$ the result is the same as the input unit and the same as with a rotation by zero. If a byte is rotated by 10 or 18



Unit's distribution $\log_2(65536)+1$ for WORD.

positions it will be the same as rotating it only two positions – the remainder after division by $\log_2(n)$ (the number of bits per unit). What this means is that the relevant values of a rotation parameter are in the range 0 … $\log_2(n) - 1$, but both in theory and in practice such a function cannot output more than $\log_2(n)$ values for a given unit.

Let's take for example the following sequence of eight (from $\log_2(256)$) bytes: 1, 2, 4, 8, 16, 32, 64 and 128. Any of these rotated with any parameter from 0 to a maximum possible 7 will be one of the other. This is just one of the 256/8 = 32 possible unique series of such units.

What we have is: eight bits of data, three bits of scrambling and 32 possible unique series whose index can be represented on five bits. Since these 32 values corresponding to all groups of rotation are independent of one another we obtained five bits of rotation independency.

The answer to the previous question is: the difference between the total number of bits used for encoding and the number of bits used for scrambling (8 - 3 = 5). Theoretically this translates into $\log_2(n) - \log_2(\log_2(n))$ bits of independency of rotation. So, for n = 65536 $\log_2(n) = 16$ and we can extract about $16 - \log_2(16) = 12$ bits of independency of rotation.

Theoretically, there is even more independent information because units may have special bit configurations – for example, 000…, 111…, 010101… and others, which after specific rotations with less than $\log_2(n)$, fall into a previous configuration. For n = 256 (bytes) there are 36 such sequences of rotation – instead of only 32 (2^5) computed as above – these are as follows: 30 series of 8, three series of 4, one sequence of 2 and two 'sequences' of 1.

A more effective solution would be a table of translation built up so that each value from the table corresponds to the whole $\log_2(n)$ possible rotations of a unit. It is obvious that in this case the distribution of the units over the table values is constant, with the exception of the special bit configurations. Furthermore, these tables have certain properties (like symmetry) which may be used to simplify their implementation.

As an alternative to translation tables, which become impractical for large unit types (like DWORD), there are several functions that can be used in order to compute the invariant values on the spot. A good example is the cubic sum of all $\log_2(n)$ rotated positions modulo n. Note that such functions can be used to generate a good translation table.

XOR+ADD FUNCTIONS

Taking advantage of the carry bit used in ADD when this function is combined with XOR it results in a pretty good encryption method. The combinations may reside on several layers of such functions as well as on key mangling, for example:

1. $\mathtt{ADD}(A_i,K_0);\ \mathtt{XOR}(A_i,K_1);\ \mathtt{ADD}(A_i,K_2);\ \mathtt{ADD}(A_i,K_3);\ ...$
2. $\mathtt{XOR}(A_i,K_0);\ \mathtt{ADD}(K_0,K_1);\ \mathtt{ADD}(K_1,K_2);\ ...$
3. $\mathtt{XOR}(A_i,K_0);\ \mathtt{ADD}(K_0,K_1);\ \mathtt{ADD}(A_{i+1},K_2);\ ...$

For the second case when only a single signature is searched (e.g. we need to check if a single plaintext signature is contained within the encrypted data) an obvious solution is to perform XOR from the current cipher's unit index with the first n units of the signature, where n must be at least the number of keys used for encryption. From this we get:

$$X_0 = A_i \wedge a_0,\ \ldots,\ X_j = A_{i+j} \wedge a_j,\ \ldots,\ X_n = A_{i+n} \wedge a_n$$

where ^ denotes XOR operation, X_0, …, X_j, …, X_n are the results after XOR, i is the current index in cipher buffer, A_i and a_j are the cipher and plaintext units, correspondingly. On X set the polynomial reduction approach may be used in order to deduce the K keys. Note that if the encryption used m keys, where m < n, the K_{m+1}, …, K_n keys will be all zero, which represents a good match confirmation.

However the problem of the carry bit remains when checking against multiple signatures or when we deal with a situation similar to the other two cases. For this we make use of a 'trick' based on the property of the least significant bit in a unit – this bit is never affected by transportation, instead it is the first generator of transportation. The opposite of this phenomenon is found at the most significant bit, which is affected by the previous carry bit and it generates overflow instead of influencing the least significant bit. For example let's see these properties on byte ADD represented in binary:

```
11010111 (215) + 00000001 (1) = 11011000 (216)
```
a simple addition, no influence on bit 0

```
11010111 (215) + 01000001 (65) = 00011000 (24)
```
addition with overflow, bit 0 remains the same

```
11010111 (215) + 01000000 (64) = 00010111 (23)
```
addition with overflow, bit 0 is unaffected by carry.

This means that the ADD and SUB functions work exactly as XOR for this particular bit; they are practically one and the same function, disregarding the transportation bit generated by ADD/SUB. Represented in binary this is:

```
0 ^ 0 = 0 + 0 = 0 – 0 = 0        0 ^ 1 = 0 + 1 = 0 – 1 = 1
1 ^ 0 = 1 + 0 = 1 – 0 = 1        1 ^ 1 = 1 + 1 = 1 – 1 = 0
```

Based on this fact and using the basic cryptanalysis described in the previous article, the suggested approach consists of extracting the least significant bit of each unit and performing XOR on every two consecutive bits, thus from n units resulting in n - 1 bits which are independent of any combination of XOR and ADD/SUB layers.

We shall call that ring0 approach which is the approach of bit 0 and has only two positions. This may be extended by adding ring1, ring2, …, ring_n, which have 4, 8, …, 2^n positions to obtain exactly $(n/2^0 – 1) \cdot 2^0 + (n/2^1 – 1) \cdot 2^1 + … + (n/2^{m-1} – 1) \cdot 2^{m-1} = m \cdot n – 2^m + 1$ bits from n units represented by m bits when n is large enough ($n \geq 2^m$) – but that is a subject for a different discussion.

This takes care of the first example; next we shall generalize the solution for the other two. Key scrambling with only one additional key has a recurrence interval of maximum two bits, when the least significant bit of the key modifier is one. To extract independent information we simply XOR every two consecutive even and odd bits respectively, obtaining n - 2 bits from n units.

TRANSLATION FUNCTIONS

These functions are such that, for any given A_0 different from A_1 we have $f(A_0)$ different from $f(A_1)$, and $f(A_0)$, $f(A_1)$ will remain constant during the process of encryption.

A typical example is a table chosen at random which is used to encrypt text. The letter being translated is replaced by the corresponding letter from the second table based on its position in the first table. For the sake of simplicity we chose the first table as the alphabet.

Table 1	ABCDEFGHIJKLMNOPQRSTUVWXYZ
Table 2	BWJQMVHXDPTYFLURSACIKNEZOG

Ciphertext: OKKKRDM CBDQ IXM VKLLO YDIIYM HDAY
Plaintext: YUUUPIE SAID THE FUNNY LITTLE GIRL

The ciphertext corresponds to the plaintext passed through the table of translation. A more general case is where several layers of functions are applied on every unit of data being encrypted which in total result in a single translation function.

In this case what remains independent of translation is the relative distance between two repeating units. For the above example the distance – excluding spaces – from the first letter (Y/O) to the second occurrence is 17. The R/A letters in plaintext/ciphertext do not offer any information of this kind. The bottom line is that the text may be represented as an array of displacements of the repeating units, disregarding every first occurrence of them – and this case covers the one frequency units. For our example: [0], [0], [3], [6], [11], [0], [17], [10], [9], [0], [3], [10], [5], [4].

On text data the displacements have a tendency to be relatively small, whereas there are many cases in which one unit of data may appear at a relatively high displacement. Imagine a null character followed by a long zone of some repeating non-null characters, then followed by another null character. For that we transform the displacements to a simpler and more effective form that consists of the number of unique units encountered between the two occurrences of the same unit. For the given example: 0, 0, 3, 6, 9, 0, 11, 9, 8, 0, 2, 8, 4, 4.

Still the most important advantage of this last representation is the fact that these numbers can be represented as units of cipher/plaintext, since the maximum number of unique units that may appear between two consecutive occurrences is n - 1, where n is the cardinal of the units' set. From our example we express the information independent of translation as follows (this time the indexes have meaning only in the alphabet, 0 corresponds to A, 1 to B, etc.): AADGJALJIACIEE.

However, besides the relatively high processing time required there is another limitation to this approach, namely the probability of repetition. High entropy data represents the worst case on which the largest block of data is required in order to perform a signature check. And that is n + m units, where n is the units' set cardinal and m is the minimum units required for signature analysis.

The problem mainly occurs for high values of n like 1<<16 = 65536 in the case of 16-bit units or 1<<32 = 4G in the case of 32-bit units. Fortunately assembly code in general and especially RISC and 32/64-bit code does not have such high entropy, so in fact less data is required, but this approach is inadequate even for 32+ bit translations and one should consider breaking the problem down into simpler ones.

BOTTOM LINE

Bit operations like XOR and ROR/ROL have certain properties which work favourably for cryptanalysis. Win32.Bagle.{M-Q,S}@mm file infectors use a random number of layers of four instructions, all operating on bytes: ROL, ROR, XOR with random constant keys and ROL with linear variation key. The first three instructions in any combination and in any number of layers always stack up in two instructions: a single ROL or ROR and a single XOR. However, the linear variation key ROL cycles every 8/4/2 bytes. So if one performs an XOR between two consecutive qwords reducing the XOR key one remains with eight bytes – dependent only on rotations – which describe the variation of rotation. Only two of these bytes are required to deduce the base rotation key and the rotation increment.

The method of cryptanalysis described here increases the number of encrypted viruses that are able to be identified to close to the limit, although it is intended as an alternative solution to existing virus identification methods.

SPOTLIGHT 1

CHECKVIR ANTI-VIRUS TESTING AND CERTIFICATION

Dr Ferenc Leitold
Veszprém University, Hungary

Certification programmes provide a useful independent assessment of the capabilities of software products. Anti-virus products present a unique challenge for testers – the nature of viruses and anti-virus products being such that, even were a product to remain static the viruses it must detect will not, thus making regular re-testing essential. In this article Ferenc Leitold introduces the CheckVir anti-virus testing service and certification programme, and in the following article Matt Ham provides an overview of Virus Bulletin's own VB 100% testing procedures.

It is important for all software testers and quality engineers to test their programs in as many different environments as possible, with numerous input combinations. In the case of anti-virus products this task is more difficult because the products change very rapidly. Anti-virus software usually includes several tens of thousands of detection and disinfection algorithms, which should be tested regularly on a large number of virus samples and, of course, on non-virus objects as well.

The *CheckVir* project has provided a regular anti-virus testing service since April 2002, and in January 2004 a monthly anti-virus certification process was started.

AIMS OF THE PROJECT

The *CheckVir* project aims to provide clear, accurate and reliable testing of anti-virus products. A number of rules were set out at the beginning of the project. Some of them are as follows:

- Infected objects must be made by replicating the virus – which proves the 'viral property' of infected objects.
- No application other than the anti-virus software will be installed on the test platform.
- All available service packs and updates of both the operating system and the anti-virus product must be installed.
- Every test must be repeatable.

The main goals of the testing and certification procedures are to decrease the number of bugs in the products, and to minimise the number of problems related to anti-virus products. Problems and bugs are distinguished as follows: a problem is when a new feature needs to be developed into the product – for example, if an anti-virus product cannot detect a virus or disinfect it. The presence of a bug means that the product does not behave correctly – for example, if an anti-virus product informs the user that a particular virus has been removed, but the cleaned program file is unable to run.

REGULAR TESTING

As part of the *CheckVir* project's regular anti-virus testing service tests are executed monthly on different platforms. Regular tests are based on virus detection and disinfection. Scanning is tested both on demand and on access, and email scanning of both incoming and outgoing traffic is also put to the test. Other tests include speed tests, heuristic tests, testing of packed objects and testing of the storage area of email clients.

CheckVir's regular testing is a powerful aid for developers in identifying the following problems that may be found in anti-virus products:

- The anti-virus software is able to detect a virus, but does not deal with unusual cases (e.g. the infected file is too small or too big where there is an error in the virus).
- The behaviour of at least two versions of an anti-virus product developed by the same company and working with the same engine are different (e.g. the *Windows Me* version of an anti-virus product can detect a virus, but the *Windows XP* version of the same product is unable to detect the same virus in the same sample).
- The behaviour of at least two scanning methods of an anti-virus product using the same engine and database are different (e.g. the on-demand scanner can detect a virus, but the on-access scanner of the same product is unable to detect the same virus in the same sample).
- The behaviour of the product's disinfection capability is different when using two different versions of the same anti-virus product (e.g. versions for different platforms).
- The behaviour of the product's disinfection capability is different when using different scanning methods (on demand and on access).
- The anti-virus software is able to detect a particular virus, but only in some samples (e.g. in the case of polymorphic and macro viruses).
- The anti-virus product does not wipe all virus-related macros correctly from a document.
- The anti-virus program is unable to distinguish between similar viruses. In some instances, the program makes mistakes during the disinfection procedure.

- The anti-virus program is able to disinfect a particular virus correctly, but the disinfected file may not be able to be executed from some samples.
- Other functional problems (e.g. the anti-virus software hangs during the disinfection procedure for a particular virus).

CHECKVIR AV CERTIFICATION PROGRAMME

The *CheckVir* anti-virus certification programme includes all products that are submitted for the regular *CheckVir* testing service. Currently there are three levels of certification:

- *Standard level*. Here, the products' virus detection capabilities (alone) are examined. The anti-virus products must find all of the virus samples in the test set. The products must provide on-demand and on-access scanning facilities, and the results of the tests of these must be the same.
- *Advanced level*. Here the products' virus detection and removal capabilities are examined using on-demand and on-access scanning. Products must meet all the conditions required for the Standard level, and they must repair all of the infected objects, with the following conditions:
    - If, theoretically, it is possible to repair the whole original object correctly (bit for bit), then the repaired object must be identical to how it was prior to infection.
    - If it is not possible to repair the whole original object correctly (bit for bit), but it is (theoretically) possible to repair the object with minor changes and with the functionality of the repaired object restored to the same as it was prior to infection, then anti-virus products must repair the object in this way: the functionality of the repaired object must be the same as it was before infection.
    - In the case of macro viruses, the macros in a repaired document and the macros that were stored in the document prior to infection must be the same (in both name and content). After a notification to the user the anti-virus product may delete all of the macros from the infected document.
    - In the case of boot viruses, infected sectors must be the same after disinfection as they were prior to infection. Data stored in the boot sector (e.g. partition information) or in the master boot record may not be changed during disinfection. If it is not possible to restore the original master boot record then, after notifying the user, the anti-virus product may generate a standard master boot record.
- *Mailscanner*. In this case anti-virus products must provide an email scanning service. Both incoming and outgoing infected messages must be identified and anti-virus products must disinfect the email or block the corresponding traffic. This means that a user cannot send or receive infected email. The Mailscanner certification is independent of the Standard and Advanced levels of certification.

The virus test set used for certification is based on viruses published on the WildList prior to the test. At least 80 per cent of the set includes viruses that are published in the last three issues of the WildList. A maximum of 20 per cent of the set may include any virus published on any WildList. The list of viruses that will be included in the test set is published at the beginning of the month on the *CheckVir* website. The anti-virus developers then have about 10 days to upgrade their products before the deadline for product submission. This means that the certification procedure does not deal with the latest viruses.

CERTIFICATION RESULTS

In the first half of 2004 six certification procedures were carried out. In one month the certification process was executed on two platforms (client and server) – in this case anti-virus developers submitted two products. The following table show the summary of the certification results:

AV developer	**No. of products submitted**	**No. of Standard certifications**	**No. of Advanced certifications**
Grisoft	7	7	-
Softwin SRL	7	6	-
ID Anti-Virus Lab	5	5	-
Computer Associates	7	2	5
F-Secure Ltd.	7	7	-
Kaspersky Lab.	7	7	-
Network Associates	7	7	-
Eset Software	7	7	-
Norman ASA	7	5	-
Panda Software	7	7	-
Trend Micro	7	1	6
VirusBuster Ltd	7	5	2
MicroWorld Technologies	2	-	2

The results of each certification procedure are published on the *CheckVir* website, http://www.checkvir.com/, in the month following the testing process.

The CheckVir anti-virus testing service is free for the first test of any AV company. More information can be found at http://www.checkvir.com/.

SPOTLIGHT 2

THE VB 100% AWARDS: AN OVERVIEW

Matt Ham

Although *Virus Bulletin* has carried out comparative reviews of anti-virus products for many years, it was not until 1998 that the VB 100% award made its first appearance (see *VB*, January 1998, p.10). In hindsight, the choice of '100%' as part of the name of the award was, perhaps, a little ill-considered – this part of the name having been responsible for many of the misconceptions about what a product must do to gain a VB 100% and what inferences can be drawn if a product gains or fails to gain the award. These misconceptions are at least part of the reason why this article has been written.

When *Virus Bulletin* was first published in July 1989, viruses were virtually unknown outside a small number of unlucky infectees and the even smaller number of individuals involved in the detection and removal of this relatively new threat. At that time it was still possible to possess a full set of all known viruses, which could be used to test any of the products on offer.

As time passed, however, the situation changed. Not only were the numbers of viruses increasing rapidly but also the existing viruses were becoming polarised between those which were a current realistic threat and those which were not. When testing a product, the active threats are, of course, more relevant where detection (or lack thereof) is concerned. The total number of viruses in existence is debatable, but 50,000 is possibly the lowest estimate I have seen recently. Of these, no more than 300 or so are being reported as an active threat at any one time.

These actively threatening viruses are reported to the *WildList Organization* (*WLO*) by a selection of volunteers the world over, the *WLO* collating the results and publishing them regularly. As most, if not all, readers will be aware, the resulting list of viruses is the WildList, with the viruses listed upon it commonly being referred to as being 'In the Wild' (ItW). *Virus Bulletin*'s own ItW test set of viruses – which is the basis of VB 100% testing – is derived from this WildList.

PREPARATION AND CONTENT OF TEST SETS

As mentioned, the samples used for *Virus Bulletin*'s In the Wild test set are based upon those reported to be In the Wild by the *WLO*. Until February 2004 *Virus Bulletin*'s test sets were drawn from the most recent monthly WildList available at the time of testing – which, as a result of delays in WildList publication, often meant using data that was as much as two months behind the product submission deadline. However, since the advent of the Real Time WildList (RTWL), *Virus Bulletin*'s test sets have been derived from the latest RTWL at midday GMT, two days prior to the deadline for product submission for the test – thus posing a greater and more realistic challenge for the products under test.

Where possible, samples are replicated from the reference samples provided by the *WLO*. Where these particular samples cannot be replicated, samples may be obtained from anti-virus developers. The use of verified samples is very important, because of anti-virus developers' notorious use of multiple names for the same virus or, more awkwardly, the same names for different viruses.

Replication serves two very important purposes. First, it proves beyond a doubt that the samples used are indeed viruses (since if they were not, replication would be impossible). Secondly, polymorphic viruses cannot be represented by the commonly available *WLO* samples, since these are easily available and may be targeted specifically for detection by unscrupulous developers. By replicating the viruses before they are used for testing, the exact internal structure of these polymorphic virus samples will be unknown to the product that is being tested, much like the situation when a real-world user is infected.

Many magazine reviews in the past have skipped this process and included in their test sets files which are non-infectious or detected far too easily by any scanner by dint of being in known malware collections. The results of this can be very misleading. Products which rightfully do not detect what amount to junk files are often penalised since the review author 'knows' he is testing against viruses. At the other end of the scale, products which might detect one out of every hundred samples of a polymorphic virus may be given that very sample to detect. Clearly, neither situation is ideal from either a user's or a developer's point of view.

Returning to the *VB* test sets, it is time to look at some more details of the samples. The samples in the test set may range from a simple worm, with one file only ever representing it, to a polymorphic virus which has billions of potentially variable samples. One sample of the worm in the test set will clearly be sufficient – any more would be overkill. On the other hand, several hundred samples of the polymorphic virus may need to be tested to give a good idea of a product's detection capabilities. For this reason the number of samples of each virus in our test sets varies considerably.

When calculating results, however, it is the number of *viruses* missed, rather than number of samples missed, that is of importance.

As an example, imagine a test set of 10 viruses: A, B, C and so on. Virus A is simple and requires only one sample to be tested. Virus B is much more tricky, requiring 10 samples. A hypothetical scanner which detected only the one sample of virus A would have a detection rating of 10 per cent – having detected one out of the 10 viruses. A second scanner which detected only the 10 samples of virus B would have an identical 10 per cent detection rating. Scanner B may have detected *more samples*, but overall it has shown itself to reliably detect only *one* of the 10 viruses on test. Finally, a third scanner which detects only one sample of virus B would receive a detection rating of a mere one per cent – it detects one tenth of the samples in one tenth of the viruses.

The net result is that the numbers of missed samples do not necessarily equate directly to percentages in the final results – something which has raised numerous queries in the past. As a result, however, each virus is given equal weighting in the test results, be it simple or fiendishly hard to detect. After all, a user is unlikely to be consoled by the fact that he is infected by a virus because it was difficult to detect.

There are some final provisos in the samples used, which aim to make these files as similar as possible to those a scanner will encounter in real-world situations. The samples used always have the same extension as found in active samples. For example, EXE files are scanned as EXE files, rather than being renamed as VXE for safer storage. Many scanners filter detections through means of extensions, so the preservation of these is important. Similarly, where viruses which infect floppy boot sectors are concerned, real floppy diskettes are used rather than using disk images. On many platforms, especially *Windows NT*, the differences in the behaviour of the anti-virus product between being faced with an image or a real diskette are considerable.

ItW File		Macro		Polymorphic		Standard	
Number missed	%	Number missed	%	Number missed	%	Number missed	%
0	100.00%	4	99.90%	1	99.89%	1	99.82%
0	100.00%	12	99.82%	437	98.50%	2	99.90%
5	98.96%	0	100.00%	0	100.00%	14	98.96%
0	100.00%	0	100.00%	1	99.92%	0	100.00%
0	100.00%	0	100.00%	0	100.00%	2	99.82%
0	100.00%	4	99.90%	180	91.24%	11	99.64%
0	100.00%	8	99.80%	60	95.79%	13	99.40%
0	100.00%	0	100.00%	0	100.00%	0	100.00%
0	100.00%	0	100.00%	600	89.13%	14	99.15%

The numbers of missed samples do not necessarily equate directly to percentages.

AWARD QUALIFICATION

Now we can look at what a product must do to earn a VB 100% award. There are two fundamental tests involved in the VB 100% award: the detection of 100 per cent of the viruses in the ItW test set, and no detection of infections in the test set of clean files.

This seems simple enough, but confusion is possible even at this level. The test sets used for review purposes are not restricted to the ItW set – the macro, standard and polymorphic test sets contain a host of viruses which range from samples that are purely of academic interest, to samples of viruses that have only just left the ItW test set. As far as the VB 100% award is concerned, these other samples are not taken into consideration, so the award should in no way be assumed to relate to 100 per cent detection of *all* viruses tested, let alone all viruses in existence.

What constitutes a detection is no longer as clear cut as it once might have been. Initially, on-demand scanning with a command-line version of the product on test was deemed sufficient. This has now expanded to include both testing of GUI-based applications and the requirement for detection in real time when a file is accessed. These two methods of scanning, referred to as ‘on demand’ and ‘on access’ respectively, are sufficiently different that they must be considered separately.

ON-DEMAND TESTS

One unusual feature of *VB* testing is that products are run in default mode as far as possible. This equates to using out-of-the-box settings, and choosing the default or manufacturer-recommended settings wherever a choice is offered. There are two exceptions. These exceptions do not change the detection functionality of the on-demand scanner in any way, but do make testing either simpler or, in some cases, possible.

First, the logging options are adjusted so that a log is produced and not truncated – this is vital so that results can be extracted after the test is complete. Secondly, any on-access scanner is disabled while on-demand scanning is tested. A surprising number of products are configured so that an on-demand scan also triggers on-access testing of each file. Depending on how the two scans are configured this can result in zero detections, the on-access scanner blocking the on-demand scanner from inspecting infected files, or detection which is more indicative of the on-access than the on-demand scanner.

In an ideal situation detection of a malicious file is considered to occur if the scanner gives any form of malware-related warning for a particular file in its log. This can run the whole gamut from a definite virus name and description, all the way to a low priority warning that the file is 'suspect'. Various scripts are used to extract the data from the log files into a more manipulatable form, the end product being a list of missed files – hopefully, for the developer, containing only a reference clean file.

Unfortunately, however, the ideal situation does not always present itself. In some cases logging results in non-parsable files or causes scanner instability due to the size of the log created. In such cases deletion is the preferred scan option – which means having to make an exception to the use of default options. Any files that remain after the scanning procedure count as misses. It is clear that this process will give easily analysed results, so some readers might wonder why I burden myself with the additional task of parsing logs.

There are several reasons, not the least of which is that developers may wish to see the logs that are produced during testing. A log is far more useful than a mere list of files in a directory. The logs also tend towards less variability in the treatment of infected, suspicious, disinfectable or beyond-hope files. A scanner set to delete might treat each of these classes of file in a different fashion – making deletion a great deal more prone to confusing results. For this reason, where deletion is used it is usually combined with logging and visual inspection of results – which not only makes the process longer but also means that there is a greater need to (re)check the results than when parsing logs.

In some software disinfection is the default – and only – action available for on-demand scanning. In such cases CRC checks are used to determine which files have been disinfected. In these cases the on-screen reports generated by the product are also used extensively to double-check the remaining files, and the treatment of non-disinfectable files must be noted carefully. This is a last resort, used only with the least controllable (and correspondingly most cursed) of products submitted for testing.

ON-ACCESS TESTS

Testing of on-access functionality is also performed as much as possible in the default mode of the software involved. The major proviso is that testing is performed by setting the software under test to block access to infected files. Each of the files in the test set is then opened automatically by a custom utility – which does not execute the file in question. Failure to block access is counted as a missed detection. The file opening utility has output available for both opened and unopened files which are used to produce results. Since these outputs are designed for test usage, the parsing of these files is usually much faster than that performed on the on-demand log files.

There are several circumstances under which special cases may occur. The most common is that a file-open alone does not trigger a detection, while copying the files in question does. In these cases xcopy or the operating system equivalent command is used to copy the test sets. Those files which are blocked from being copied are considered to be detected.

A more tricky situation occurs if there is no block option available in the software in question. In such cases deletion is the preferred scan option, with any files remaining at the end of the scan being counted as misses.

In some software disinfection is the default – and only – action available for on-access scanning. In such cases CRC checks are used to determine which files have been disinfected, and the log files of the product are used extensively to double-check the remaining files. As with the on-demand testing, in some cases there is a quite disturbing lack of control available for the functioning of the scanning product.

FLOPPY TESTING

The detection of floppy diskette boot sector viruses is similar to the processes described above, though with differences necessitated by the medium in question.

Since actual diskettes are used, these are inserted individually to be scanned on access or on demand. In both cases the operating system may be slow or unable to detect the changes of media, which will render a scanner unable to re-scan at the correct time. For this reason a blank (uninfected) 720 kB floppy is inserted between each infected 1.44 MB diskette. This change of diskette capacity is usually sufficient to persuade even the most recalcitrant operating system that something has been altered in the drive.

FALSE POSITIVE TESTS

As mentioned, the detection of infected files and diskettes is not the only hurdle a product must cross before it can achieve a VB 100% award. The tests so far would allow a product to declare any and every file a virus, thus detecting all of the viruses in the test set. However, such a product would practically be worthless since false alarms of positive detection would be produced on any uninfected file that was scanned.

Although no product has yet operated in quite such a paranoid fashion, another requirement for certification is that a product produces no false positive detections on scanning a collection of files that are known to be clean. Alerts produced in this clean set are counted as being unforgivable if a definite virus presence is declared. Cases where files a declared as 'suspicious' do not forfeit a VB 100%, though the presence of suspicions is noted in the review.

RESULTS

For an overview of past results, the best source is, of course, the *Virus Bulletin* website: http://www.virusbtn.com/. Here, there is a summary table which displays the results of the most recent comparative reviews for each platform tested and a history showing each AV vendor's performance in previous reviews. Of course the individual reviews themselves, published in *Virus Bulletin* magazine, include more exact details – including notes of any instability encountered, impressive improvements to the interface or simply peculiarities which caught my eye.

There are many tales of woe concerning products that have failed narrowly to qualify for a VB 100% award. I stress that the failure to gain a VB 100% award is not, in itself, any reason to condemn a product.

In order to use the results of these tests in any serious fashion, historic trends for a product must be examined. Most developers will concede that, for ItW viruses at least, detection is uniformly good over almost all products. Misses can occur as a result of bad luck, bad timing or an oversight in default settings in an otherwise solid product. Whether these are of relevance to an end user depends on the individual user's requirements and situation.

It is because there are such important caveats to be considered, that the *Virus Bulletin* reviews have never offered recommendations or top scorers. *VB* provides the information and the choice of product must be the end user's decision alone.

Developers of anti-virus products who are interested in submitting products for Virus Bulletin*'s comparative reviews should contact matthew.ham@virusbtn.com. There is no charge for products to be included in the* Virus Bulletin *comparative reviews. The next comparative review will be on Windows 2003 Server and the results will be published in the November 2004 issue of* Virus Bulletin*. A schedule of forthcoming comparative reviews can be found at http://www.virusbtn.com/vb100/.*

vb 2004 CHICAGO

Virus Bulletin International Conference & Exhibition

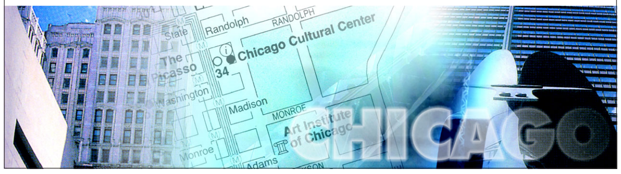
The Fairmont, Chicago, 29 Sept – 1 Oct, 2004

Join the *Virus Bulletin* team in Chicago for the anti-virus event of the year:

- 30+ presentations by world-leading experts
- Real-world anti-virus and anti-spam case studies
- Law enforcement
- Forensics
- Corporate policy
- Emerging threats
- New technologies
- Lively panel discussions
- Comprehensive exhibition
- Networking opportunities
- Reduced registration fee for *VB* subscribers
- Full programme at www.virusbtn.com

REGISTER ONLINE AT
WWW.VIRUSBTN.COM

email: vb2004@virusbtn.com; tel: +44 1235 555139



PRODUCT REVIEW

KASPERSKY ANTI-VIRUS PERSONAL 5.0

Matt Ham

Some years ago, when I first joined *Virus Bulletin*, I was handed my first product to review. Emblazoned with pictures of Michelangelo's David, the product was *KAMI*'s *AVP*. The name may be different and the pictures on the box are now of the (thankfully less naked) Eugene Kaspersky – but *Kaspersky Anti-Virus* (*KAV*) is a direct descendant of that scanner.

At the time of that first review *AVP* was the darling of the industry, the product of an almost unknown company which nonetheless detected in the same league as the larger players in the market. The obscurity has now faded, leaving a product which still detects well, but is certainly no longer an underdog.

The detection abilities of *KAV* in its various incarnations have remained steady and *Kaspersky Lab* has been the recipient of many VB 100% awards as a result. As is usual in a standalone review, therefore, readers are referred to the last few comparative reviews for information on *KAV*'s ability in its default mode.

Currently *Kaspersky Lab* produces three types of software: anti-virus, anti-hacker and anti-spam. The platforms upon which these products run are many and varied – *Kaspersky* is certainly one of the more diverse among anti-virus vendors. In addition to the standard anti-virus offerings for desktops and file-servers there are scanners available for PDAs, firewalls and groupware products. The *KAV Personal* submitted for testing is aimed towards the home user end of the market and was thus tested on *Windows XP* and *Windows 98 SE* – these both being popular operating systems in that sector. (*Windows XP* was not patched to *SP2* level, since this was only in the process of general release during the later stages of testing.)

WEB PRESENCE AND DOCUMENTATION

The hard copy documentation supplied with this version of *KAV* consists of a 96-page manual. This has clearly been updated carefully for the new product and shows none of the anachronisms which plague manuals that have simply been edited for the arrival of successive new product versions. Although it contains glossaries and various pieces of background information the bulk of the manual is devoted to the operation of the program.

The same manual is also available from within *KAV*, as a portion of the help function. Help is partially context-sensitive, in that each page has specific help associated with it, though help cannot be triggered for each option individually on a page. The vast bulk of the help available is contained within the manual, though there seemed to be frequent additions for individual functions, which were available through links within the help file.

The *Kaspersky* web presence, for English language users at least, is centred around two sites – www.kaspersky.com and www.viruslist.com. The former is at first glance very much the typical AV developer website, with information covering corporate and threat-specific news, product lines and general business contacts. Although news concerning some recent threats can be found here, there is major omission in comparison with many vendor websites – that of detailed searchable virus information. Enter the www.viruslist site, which provides exactly this sort of information. This site has a news-centred home page, which covers all manner of security issues, although virus information dominates.

A large searchable database of virus information is also available on this site, with articles linked explaining further general details. For example, if looking for information on Form, links are provided to information on many aspects of boot viruses and to a glossary of related terms. This is a very user-friendly way of providing the information, since those who need to access the explanatory articles can find them easily, without boring those who already know enough. *Kaspersky*'s ownership of the site is less obvious than it was in the past, though advertisements for *Kaspersky* products still exist, as might be expected.

INSTALLATION AND UPDATE

Unusually for a standalone review, *KAV* was supplied as a boxed set, so installation was performed directly from the CD. The product was installed on a machine connected to the Internet via an ADSL link. The CD operates an autorun, which, rather than presenting a menu, goes straight to the point in launching *KAV*'s installation.

The first instructions from the installer are to close down any other programs that are running on the machine – the usual spiel from an installer. In this case, however, users may be more likely to comply, since the notification indicates that closing other programs will mean it is less likely that a reboot will be required.

For the purposes of a first installation, *Notepad* was left open on the machine – primarily because I was writing this review at the time. This did result in a reboot being required on the system, though further installations with no applications open showed the same behaviour. With, for example, Internet connectivity software invisibly active during installation, obtaining a machine with no other programs running could be considered something of a tall order.

Next in the process is the licensing agreement. These documents tend to be much of a muchness for anti-virus products, though there were some notable differences here. For example, there is a greater emphasis than usual on some obscure methods of avoiding licensing fees. In addition *Kaspersky* does not share most AV companies' aversion to the installation of the product in nuclear power plants, and the company is even so bold as to state that the product *will* detect viruses – albeit only for 90 days.

Once the licence has been agreed to, a user name and organisation must be entered, after which additional information is displayed. This is something of a combination of a secondary licensing agreement, readme file and feature listing. Here, we learn that *KAV* supports detection in some 900 run-time compression methods and 90 static archive types. Less impressive, but showing honesty at least, is the note that uninstallation of the product may not always work the first time – with the traditional solution being offered: if it doesn't work, try again until it does.

The next prompt is for an install location. The space noted as being required is 14.7 MB – making this not the most slimline of products but certainly not oversized by current standards. After selection of a location, the installation procedure completes. Following a reboot the only sign that installation has occurred is a *Kaspersky* symbol as a tray icon, which appears with a muted plopping sound.

The first activation of the *KAV* software was performed through the Start menu, and as expected the first page encountered noted that, although on-access scanning was activated, no full scan had been performed and the definition files were 'totally outdated'. This was a fair comment since the files in question were 124 days old. The update option was chosen, which connected to *Kaspersky*'s update site quickly.

The size of update was 4 MB, though this large size did not come as a surprise with such an old initial install. In comparative reviews, where Internet-based updates are impossible due to the isolated test environment, updating of *KAV* can be quite a chore; this is certainly not the case when Internet connectivity is present.

Since the uninstall routine had been noted as being subject to problems, this was investigated further. No problems were encountered at all if the machine was rebooted straight after the uninstall application was completed. However, if other programs were installed before the reboot (in this case graphics drivers were updated), numerous problems were encountered. Several DLLs were noted as missing during the next boot and Java scripts were rendered inoperative. Clearly program files had been removed but were still being invoked. Reinstalling and uninstalling once more solved the problem.

PRODUCT OPTIONS

As with most scanners, the three primary scan methods for *KAV* are the on-demand, on-access and scheduled scans. In recent versions of *KAV* these have each been blessed with their own GUI – a situation which has had more cons than pros during my use of the applications. It seems that developers have taken note of feedback that considered this split personality to be a problem, since the newest version is now monolithic in nature. It has also received a major overhaul in look and feel, now being very much more *Windows* standard.

The GUI offers three tabbed views: Protection, Settings and Support. In more than one place it was noted that default options for the software had been improved during the product revamp, so these were examined more closely than usual. Each of these tabbed views follows the traditional left selection pane and right interaction pane for various sets of instructions within the view.

The starting view is the Protection tab, which offers in the right pane on-access status, full scan status and virus database status. Initially, only the on-access start is noted as having performed correctly.

The full scan is set up as a scheduled job at 5pm each Friday – which seems a reasonable choice. A scheduled scan a little later in the day might be preferable though – although many European companies operate short Friday working hours, this is not a universal habit! As a home product, however, this could be seen as a scan before the home-user arrives home for a weekend of surfing and email – possibly a more logical choice. Since this scan time is adjustable, of course, exact scan times are not a great issue.

The update process is not triggered on first loading the product, which was somewhat worrying – for example, this allows a user to perform the initial full scan before having updated the software. This problem was encountered not only with the CD version, with definitions included being over 100 days old, but also on the web download of the software, which was 27 days old at the time of testing.

Returning to the Protection view, the left-hand pane offers a choice of four sections. The first of these offers Scan My Computer, Scan Removable Drives and Scan Objects, while the other three each have a single entry: Update Now, View Quarantine and View Reports.

If one of the three Scan entries is chosen the result is a scan with no further choices as to configuration. The Scan Objects selection offers a choice of locations to scan – and users may browse if the selection offered is not deemed useful – but there are no choices as to the mode of scanning or action upon scanning. Likewise, the Update Now option in the left-hand pane simply updates, with no configuration options being present here.

When a scan is run the default option is for the user to decide what action should be performed upon detection of infected objects. A noise reminiscent of squealing pigs is produced when a virus is detected and in practice the choice was limited, in default mode, to disinfecting, deleting or ignoring each sample. Once the choice has been made it can be set to be applied to all following detections in the same scan.

The View Quarantine and View Reports options can be considered together as sources of information about past scans. Viewing of reports is very much standard for its type, though actions cannot be applied retroactively; the reports are for information only. The Quarantine area is somewhat novel in its approach. Files can reach the quarantine in one of two ways: either through being flagged as suspicious by heuristics or by being added directly by the user. Once in the quarantine area the files may be restored, sent to *Kaspersky* for further analysis, deleted permanently or scanned. Presumably scanning is selected when a new virus definition file has been applied, checking for files which have moved from heuristic to exact detection in the newer update.

The default starting tab of Protection allows direct access to all the major functionality of the scanner, without actually exerting any control over the specifics of what is done, for example, during a scan of files. For adjustment of functionality the next tab, Settings, must be selected.

Settings offers five views selectable from the left pane: Configure Real-Time Protection, Configure On-Demand Scan, Configure Updater, Configure Quarantine and Additional Settings. The default contents of the right-hand pane are details of the current configuration for on-access, on-demand and update functionality. These show green tick icons if the settings are those approved of by the developers, or yellow warning triangles if not.

Configure Real-Time Protection is the on-access portion of the configuration. When choosing to configure this type of scan the interface will come as quite a surprise to those used to *KAV*'s usual myriad options. There are effectively three settings: the default, a more secure set of scanning rules and a set of options stressing speed over absolute security. The first two of these display the approved green tick, while the speedier option shows a yellow triangle of danger.

By default, the on-access scanner is targeted at 'all potentially infectable files', though outgoing mail, self-extracting archives and some other archives are not investigated. The maximum level of on-access security adds detection within self-extracting archives and outgoing mail.

The minimum level of on-access protection has only one difference from the default – this being that file types are determined by extension. This is certainly an easy way of cutting down the overhead of checking files for, for example, MZ headers to ascertain that they are executable – though I would say this was somewhat undesirable by, in effect, applying a list of extensions which will be scanned to the exclusion of all others. Since extensions have caused *Kaspersky* some trouble in the past, the yellow triangle seems appropriate as a warning to customers.

Apart from the changing of security levels, the action on detection may also be set here. The default is to block access and prompt for further action, though automatic 'recommended actions', deletion or logging may be selected. The default settings may be restored with use of a hot spot.

More control over the on-access scanning functionality can be exerted through the 'troubleshooting' link. Exclusions can be initiated here, for example. Scanning may also be set to time-out for excessive scan times, boot sectors may be excluded from scanning and on-access scanning removed for files, mail and scripts independently of the others. In extreme cases the on-access scan may be disabled completely. Somewhat strangely, if all on-access scanning is removed, the status of scanning in the Settings view is still awarded only a yellow triangle of warning, and the Protection level can be declared as High or Recommended. In the Protection view, however, the red circle of wrath is directed towards such a foolish choice. After fiddling about the 'restore default settings' function seemed to work perfectly.

On-demand scanning is similar in theory to the on-access scanning, with three levels of scanning available, actions on detection configurable and a set of troubleshooting options. Added to these options is the ability to invoke the scheduler for on-demand jobs. The difference between the Recommended, the default scan and Maximum Protection is not fully described – only stating that scans are more thorough if the latter is selected. High-speed scanning, on the other hand, opts not to scan archives, mail databases and 'objects executed during system startup' – the documentation does not elaborate further.

Troubleshooting options here are very different from those offered by the on-access scanner, though the exclusion list is present once again. Boot sectors, system memory, compressed executables, archives, startup objects and email databases may each be excluded from scanning regardless of other settings. It is also possible to remove the requests for passwords when scanning password-protected objects. Asking for these passwords is, in its own right, a feature which is new to me. Most products are content to break password protection where this can be done easily, while ignoring any files where there are any difficulties due to password protection.

The configuration options for the scanner are exhausted at this point, and I was somewhat surprised by their limited nature. This version of *KAV*, however, is much more the novice single-user version than the more commonly investigated *Personal Pro* variant. More general configuration options are available on the Settings tab in addition to those controlling the scanner directly. Configure Updater is probably the most important of these. There is, in fact, a control specified here which offers control over levels of scanning, though it will not show up elsewhere.

The general update mechanism downloads the standard *Kaspersky* definition files by default. *Kaspersky* offers two additional sets of databases, varying in the extent to which non-replicative malware is included in the definitions. The most extensive of these can be obtained only by direct download, but the one that adds Trojan information, rather than such things as jokes, may be selected as the default download in the Updater settings view.

Other settings here are less interesting, though probably more useful in general – update frequency and location being among those items defined here. Internet settings for proxies are obtained directly from *Internet Explorer* settings – although it was not tested, this could cause problems where a proxy is required but a user is browsing with alternative, non-*Microsoft*, applications.

Quarantine settings are the penultimate view, offering controls as to maximum size of quarantine and maximum duration of stay in quarantine before items are deleted. By default these have no limits. By default the quarantine is not rescanned when definitions are updated. With the unlimited settings on the duration of objects' stay within the quarantine this could potentially be better set as a default to scan such objects.

Finally come the Additional Settings. These are the odds and ends which do not fit happily elsewhere. Here, pop-up messages and sounds may be turned off, logging may be configured, and the software disabled as a start up application. Finally passwords may be applied to *KAV*, preventing unauthorised changes in configuration or indeed use of the program.

CONCLUSIONS

In past comparative reviews *KAV* has generally been criticised for three factors: updates, interface and default settings. Of these the method of updating was probably the one that caused the most pain, since the nature of comparative reviews requires that updating be performed manually. This is a function of the testing environment, rather than a problem likely to be encountered by real-world users. While manual updating requires the transfer of numerous files, the automatic version used for these tests was far more simple.

The interface was a problem likely to present itself to all users. However, the interface of this version is easy to handle from the start and can be considered a marked improvement from this perspective. There are more control options available in *KAV Personal Pro* – so that version may be worth considering for more advanced users, despite a less friendly interface.

The recommended settings for *KAV* are very much proposed as the 'correct' settings, with warning triangles being produced if settings are experimented with. This is both likely, and presumably designed, to dissuade users from making changes to the configuration without being confident of the results. This is undeniably a good thing, so long as the default settings are appropriate. In terms of scanner settings these tests revealed no major problems. All files are scanned, which removes one of *KAV*'s past problems: that of certain extensions being missed.

There were concerns about the settings for updates and the initial scan. Since updates occur automatically only at three-hourly intervals, no update is performed by default when a machine is booted. What is more, there is no provision for an update on boot. If, for example, a user arrives home from holiday and powers up their machine, the virus definitions might be as old as two weeks for up to three hours. Given some of the fast-spreading worms of recent months this would be ample time to infect the machine before the update is triggered.

Considering the product as a whole, many of the issues put forward during past comparative reviews have either been rendered null or proven to be an issue only under test conditions. The product has become simpler to use, though some small niggles remain concerning updating. *Kaspersky* has a good record for constant change and improvements to their scanners, so it is entirely reasonable to expect that this issue will be addressed sooner rather than later.

Technical details

Product: *Kaspersky Anti-Virus Personal 5.*

Test environment: Identical 1.6 GHz Intel Pentium machines with 512 MB RAM, 20 GB dual hard disks, DVD/CD-ROM and 3.5-inch floppy drive running *Windows XP Professional SP1.*

Athlon XP1400+ machine with 512 MB RAM, 20 GB dual hard disks, DVD/CD-ROM and ADSL Internet connection running *Windows 98 SE.*

Developer: *Kaspersky Lab*, 10 Geroyev Panfilovtsev St, 125363 Moscow, Russian Federation; email sales@kaspersky.com; websites: http://www.kaspersky.com/ http://www.viruslist.com/.

END NOTES & NEWS

The High Technology Crime Investigation Association International Conference and Expo 2004 takes place 13–15 September 2004 in Washington, D.C., USA. The conference aims to provide training for all levels of the cyber-enforcement community from security specialists to law enforcement personnel. See http://www.htcia2004.com/.

The ISACA Network Security Conference will be held 13–15 September 2004 in Las Vegas, NV, USA and 15–17 November 2004 in Budapest, Hungary. Workshops and sessions will present the program and technical sides of information security, including risk management and policy components. Presentations will discuss the technologies, and the best practices in designing, deploying, operating and auditing them. See http://www.isaca.org/.

FINSEC 2004 will take place in London, UK on 15 and 16 September 2004, with workshops taking place on 14 and 17 September. Case studies and discussion groups will cover a range of topics including: Basel II/ IAS and IT security, prevention of online fraud and phishing scams, integrating technologies into a secure compliance framework, virus and patch management, and outsourcing IT security. For full details see http://www.mistieurope.com/.

The 14th Virus Bulletin International Conference and Exhibition, VB2004, takes place 29 September to 1 October 2004 at the Fairmont Chicago, IL, USA. For details of the conference, including online registration, conference brochure and the full conference programme (complete with abstracts for all papers and panel sessions), visit http://www.virusbtn.com/.

Compsec 2004 will take place 14–15 October 2004 in London, UK. The conference aims to address the political and practical contexts of information security, as well as analysing leading edge technical issues. For details see http://www.compsec2004.com/.

RSA Europe takes place 3–5 November 2004 in Barcelona, Spain. More information, including track sessions and speaker details are available from http://www.rsaconference.com/.

The 31st Annual Computer Security Conference and Expo will take place 8–10 November 2004 in Washington, D.C., USA. 14 tracks will cover topics including wireless, management, forensics, attacks and countermeasures, compliance and privacy and advanced technology. For details see http://www.gocsi.com/.

The 7th Association of Anti-Virus Asia Researchers International conference (AVAR2004) will be held 25–26 November 2004 at the Sheraton Grande Tokyo Bay hotel in Tokyo, Japan. For details see http://www.aavar.org/.

Infosec USA will be held 7–9 December 2004 in New York, NY, USA. For details see http://www.infosecurityevent.com/.

Computer & Internet Crime 2005 will take place 24–25 January 2005 in London, UK. The conference and exhibition are dedicated solely to the problem of cyber crime and the associated threat to business, government and government agencies, public services and individuals. For more details see http://www.cic-exhibition.com/.

The 14th annual RSA Conference will be held 14–19 February 2005 at the Moscone Center in San Francisco, CA, USA. For more information see http://www.rsaconference.com/.

The E-crime and Computer Evidence conference ECCE 2005 takes place at the Columbus Hotel in Monaco from 29–30 March 2005. A reduced daily registration rate of 150 Euro per delegate applies until 21 November 2004. For more details see http://www.ecce-conference.com/.

The 14th EICAR conference will take place from 30 April to 3 May 2005 either in Malta or in Edinburgh. Authors are invited to submit non-academic papers, academic papers and poster presentations for the conference. The deadline for submissions are as follows: non-academic papers 26 November 2004; academic papers 14 January 2005; poster presentations 18 February 2005. For full details see http://conference.eicar.org/.

The sixth National Information Security Conference (NISC 6) will be held 18–20 May 2005 at the St Andrews Bay Golf Resort and Spa, Scotland. To register interest in the conference before registration opens see http://www.nisc.org.uk/.

SUBSCRIPTION RATES

Subscription price for 1 year (12 issues) including first-class/airmail delivery: £195 (US$310)

Editorial enquiries, subscription enquiries, orders and payments:

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

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

VB Spam supplement

CONTENTS

NEWS & EVENTS

CHINESE INTERNET FIRM SUSPENDED

Chinese Internet company *Sohu.com* will be subject to a one-year suspension as a supplier of mobile phone picture services as punishment for carrying out unsolicited marketing. The suspension was handed out by state-controlled mobile phone company China Mobile for an incident in June 2004 in which *Sohu.com* sent out 1,374 solicitations for its picture messaging service without approval. The suspension takes effect on 1 September.

VIRGINIAN LAW IS CONSTITUTIONAL

A judge in the US has ruled that Virginia's anti-spam law is constitutional. Back in May this year *VB* reported that the counsel representing a man charged with spam offences had called for the case to be dismissed since, they argued, the Virginian anti-spam law violated the federal Commerce Clause and the First Amendment (see *VB*, May 2004, p.S1). Last month, however, Loudoun Circuit Court Judge Thomas D. Horne upheld the constitutionality of the law.

Horne ruled that the subject heading is the only part of an email that can be argued to contain any content protected by the First Amendment. He said that the law does not penalize people who wish to remain anonymous in their emails, but, "It is an enforcement mechanism to sanction abuses of private property interests through purposeful falsifications of routing information."

While the defence attorneys representing the alleged spammer Jeremy Jaynes and his co-defendants argued that the spam law violated the due process rights of their clients because the provisions of the law are "impermissibly vague", Horne found the felony provisions to be constitutional. Horne commented that the courts have struggled with how to apply the history of the Commerce Clause to the Internet, which he described as "a communication medium that knows no geographic or political boundaries". Defence attorneys argued that the spam law violated the Commerce Clause because it controlled the commerce of other states; however Horne ruled: "The statute is not content-based, vague, or offensive to interstate commerce." Jaynes and his co-defendants will stand trial in early September.

PHISH IT YOURSELF

Do-it-yourself phishing kits are available free of charge to anyone surfing the Internet according to researchers at *Sophos*. The DIY kits include the graphics, web code and even all the text any would-be phisher would need to construct a fake online banking website, as well as the spamming software to send the emails as bait. Worryingly, while *MessageLabs* reports that it is currently intercepting around 250,000 phishing-related emails every month, a survey carried out by anti-spam firm *MailFrontier* indicated that 28 per cent of the US customers it surveyed were fooled by phishing scams – even by some of the earliest, most unsophisticated and highly publicised scams.

EVENTS

The Organisation for Economic Cooperation and Development (OECD) will hold a workshop on spam from 8–9 September 2004 in Busan, Korea. The objectives of the workshop, which is sponsored by the Ministry of Information and Communication in Korea, are to build on the results of the Brussels Workshop on Spam, held in February 2004, and attempt to explore some of the issues and problems in greater detail. See http://www.oecd.org/.

INBOX East takes place 17–19 November 2004 in Atlanta, GA, USA. Building on the INBOX West and Email Technology Conference (ETC) held in June this year, this event will feature over 50 sessions across five tracks: Systems, Solutions, Security and Privacy, Marketing and 'The Big Picture'. See http://www.inboxevent.com/.

SPOTLIGHT

ISIPP: A BRIDGE OVER TROUBLED WATERS

Anne P. Mitchell, Esq.
Institute for Spam and Internet Public Policy, USA

The Institute for Spam and Internet Public Policy (ISIPP) was set up to provide a bridge between the email-receiving and email-sending communities. Anne P. Mitchell, Esq., president and CEO of the ISIPP, explains the some of the Institute's goals and initiatives. Anne is also a Professor of Law at Lincoln Law School of San Jose where she teaches 'Spam and the Law', and a member of the Asilomar Microcomputer Workshop planning committee.

The flip side in the fight against spam is the effort to ensure that users get their baby (all of their email), but little of the bath water (spam). IT managers the world over often find themselves facing this Hobson's choice: protect their mail servers and users' inboxes from spam, or ensure that users receive every piece of email that they actually want.

The Institute for Spam and Internet Public Policy (ISIPP) was formed in 2003 to provide a much-needed bridge between the email-receiving and email-sending communities, and to help ensure that wanted email is delivered, while unwanted email is not.

The ISIPP (http://www.isipp.com) works towards these twin goals in a number of ways, and through a number of initiatives, including industry working groups, conferences, and instructional materials. Nearly all of these are geared towards helping IT managers of both sending and receiving systems to navigate the myriad of requirements and choices with which they are now faced.

Among other things, the ISIPP hosts the Email Processing Industry Alliance (EPIA) and the International Council on Internet Communications (ICIC); maintains an email senders' accreditation service and database (IADB); offers practical information such as how to comply with CAN-SPAM; provides assistance with trademarking domains and suing domain spoofers; and sponsors both national and international conferences focusing on spam and the law.

EMAIL PROCESSING INDUSTRY ALLIANCE

The Email Processing Industry Alliance is a cross-industry alliance of email service providers, ISPs, online marketers, and spam filtering companies, all working together, and making industry recommendations to help ensure that they deliver only the email that users want, and none of the email that users do not want. There are around two dozen members from both the sending and receiving communities, which include: *Innovyx*, *Silverpop*, *YesMail*, *SpamAssassin*, *MSN*, *AOL*, *MailShell*, *Outblaze*, *Digital Impact*, *Ironport* and *Messagegate*. Every EPIA member is dedicated to the same goal, and subscribes and adheres to the highest standards of email processing management. EPIA members work together to help each other and the industry to achieve both maximum email deliverability, and minimum spam.

INTERNATIONAL COUNCIL FOR INTERNET COMMUNICATIONS

Increasingly people are coming to realize that spam is a global problem, requiring global cooperation. The International Council for Internet Communications (ICIC) is a relatively new ISIPP initiative, which was announced at the end of the ISIPP's 2004 International Spam Law and Policies conference.

ICIC is a private industry group consisting of high-level executives and attorneys who provide information and context regarding the countries in which they have experience and practice, and each of whom has connections to one or more leaders in the industry in that country. Charter members include: Lindsay Barton of the Australian National Office for the Information Economy; British MP Derek White; Furio Ercolessi of the Italian ISP Spin.it; attorney Jean-Christophe Le Toquin of *Microsoft*'s Europe-Middle East-Africa office; Suresh Ramasubramanian of India-based ISP *Outblaze*; attorney and advisor to the government of Taiwan Christopher Neumeyer; and Canadian attorney and professor of law Michael Geist. Each member of the Council participates as a private individual, and not as a representative of the institution for whom they work.

ISIPP ACCREDITATION DATABASE

The ISIPP also maintains the ISIPP Accreditation Database (IADB), which is a DNS list of the domains and/or IP addresses of email senders who either meet the ISIPP's criteria as determined by background, reference and other checks, or who are known personally to the ISIPP to meet the criteria and to be good Internet mailing citizens.

Email receivers (ISPs, spam filters, enterprise servers, etc.) check the IADB in real time, as they are receiving email from the sender, to determine such information as the sender's SPF status, the opt-in policies of the sender, whether they are listed with Bonded Sender or *Habeas*, and whether the sender participates in best practices organizations or is known to ISIPP as a 'best practices sender'. Presently the IADB is helping email receiving systems make email acceptance, delivery, and processing decisions for more than 525 million pieces of email per month, on behalf of more than 60 million email boxes. There is no charge to query the IADB.

MATERIALS

Publications available from the ISIPP includes information on CAN-SPAM compliance ('The CAN-SPAM Compliance Pack' and the free '10 Things You Need to Know About CAN-SPAM'), as well as select proceedings from its national and international spam law and spam policy conferences.

SUING DOMAIN SPOOFERS

Domain spoofing – when a spammer uses someone else's domain in the 'from' line of the spam email – causes great hardship for the owner of the domain that has been spoofed. Not only does the owner of the spoofed domain have to deal with all of the bounce messages generated by the spoofing spam run (often voluminous enough to bring down a mail server), but to add insult to injury, the fact that their domain name appears in spam may cause their own legitimate email to be blocked by spam filters around the world.

The ISIPP's latest offering is a service which will help businesses to register their domain for trademark protection. Once a domain is protected by trademark law, someone who spoofs the domain can be sued for trademark infringement – something else with which the new ISIPP service can help. By using trademark law, the domain owner can sue not only the person who pressed 'send' and injected the spam into the Internet stream, but also anyone who is benefiting from the infringing spam; this is important because it is often much easier to find and sue the vendor who is advertised in the spam than it is to find the person who pressed 'send'.

CONFERENCES

Finally, the ISIPP brings it all together for those in the anti-spam industry by sponsoring both a national conference, 'Spam and the Law', and an international conference, 'International Spam Law and Policies'. These events feature talks given by top experts regarding legal, technological and policy-based methods for dealing with the spam problem. Full audio coverage as well as handouts from the conferences are available from the ISIPP (see http://www.isipp.com/conference-proceedings.php).

The January 2004 'Spam and the Law' conference was opened by California Attorney General Bill Locker and included talks from such noted experts as Prof. Larry Lessig and Apple evangelist Guy Kawasaki.

July 2004's 'International Spam Law and Policy' conference featured guests and speakers from around the world, who talked about the international roadmap to cooperation in the fight against spam. *Microsoft*'s leading anti-spam attorney in Europe, Jean-Christophe Le Toquin, who is responsible for the company's legal action in Europe, the Middle East, and Africa, invited conference delegates to send him their spam. According to Le Toquin *Microsoft* is ready, willing and able to go after spammers in those regions, but consumer complaints are required to start the process.

Another hot topic at the conference was sender authentication. There is a fairly broad consensus that all sending sites should publish SPF records, and receiving sites should check for the same. The ISIPP's IADB is particularly useful in this area, as it both confirms (via the DNS query) whether a listed site publishes SPF, and it also has a companion database of domains and subdomains cross-referenced to the IADB-listed IP addresses which are associated with those domains. This effectively provides to the receiving mail server information as to the only IP addresses that are authorized to send email 'from' the associated domain – another check against spoofing.

The use of trademark law to protect domain names and to take action against those who spoof domains, received strong support from speaker Michael Grow, who was one of the first attorneys on the anti-spam scene when he successfully sued Sanford Wallace and his company on behalf of *AOL*. Grow told conference attendees that "use of a domain name that is identical to a mark registered by another, in a header or in the text of an unsolicited electronic communication, may be deemed an act of counterfeiting," and that "any person who uses a counterfeit mark that is substantially indistinguishable from a federally registered mark in connection with the sale, offering for sale, distribution, or advertising of any goods or services, in a manner that is likely to cause confusion, or to cause mistake, or to deceive, is liable for trademark infringement."

Finally, he offered the happy news that in assessing damages for trademark counterfeiting, a court can award the domain owner financial damages of up to three times any profits gained by the infringer, or damages caused by the infringement, along with making the spoofer pay the domain-owner's attorney fees.

As is demonstrated by the sponsoring of these conferences, and by offering a service to help domain owners to register their domains for trademark protection, and to help them put together the materials necessary to sue domain spoofers, the ISIPP is committed to helping email receivers to protect and take back their inboxes. At the same time the ISIPP is dedicated to helping legitimate email senders to do the right thing and get their mail delivered through programs and initiatives like the ISIPP Accreditation Database and the Email Processing Industry Alliance.

SUMMARY

ASRG SUMMARY: AUGUST 2004

Helen Martin

Following Paul Tenny's declaration that he believed that "time and proper research have shown that laws are not the solution to spam", Matthew Elvey kicked off this month's ASRG postings with a defence of anti-spam laws, saying that the success of Australia's anti-spam laws is proof that legal measures act as an effective deterrent to spammers. He added, "nearly everyone … knew CAN-SPAM wouldn't work because it was defective, not because laws couldn't work."

Larry Seltzer lit the proverbial blue touch paper with his comment: "SMTP authentication will make a big difference for the US CAN-SPAM Act. It should be much easier to demonstrate who is spamming what to whom." The debate that followed, on the merits or otherwise of SPF (*et al.*), was long, drawn-out and heated – so much so that eventually ASRG Chair John Levine intervened, requesting an end to the flame wars. "It's built up enough of a political head of steam that MARID will report out SPF or something like it, and some people will implement it, after which we will be able to observe what happens."

Jim Fenton announced that a BOF session on Message Authentication Signature Standards at the 60th IETF would discuss the formation of an IETF working group around cryptographic signature-based approaches to address email spoofing. He indicated that there is an associated mailing list, ietf-mailsig@imc.org, which is currently open for subscriptions.

Dave Crocker brought up the subject of Bounce-Address Tag Validation (BATV), which signs the RFC2821.MailFrom cryptographically. He directed list members to a rough description at http://brandenburg.com/specifications/draft-crocker-marid-batv-00-06dc.html. Matt Sergeant enquired as to whether there was any data on how well this technique works with current mailing list software, saying that, as far as he was aware, ezmlm checks subscribers based on MailFrom, and so this scheme would not be workable with ezmlm mailing lists.

John Levine responded, saying that ezmlm is the only list manager he has encountered that uses the envelope address. He added that he had also experienced problems with jfaxsend.com, which uses the envelope address to decide who gets to use their outgoing fax gateway. He described his 'band aid' solution as keeping an exception list of domains which need unsigned addresses – this currently comprises three entries: jfax and two ezmlm servers. He said that if, in the longer run, there is an agreed spec for a BATV signature, he didn't think it would be difficult to persuade the person maintaining ezmlm-idx to strip the signatures.

Tim Bedding said he was considering the idea of a regulatory body in the UK using fines to ensure ISPs' compliance with certain rules to tackle spam. His list of rules was as follows:

1. ISPs must ensure rapid isolation of zombie machines.
2. ISPs must perform filtering or a reasonable alternative to ensure that IP spoofing is not possible.
3. ISPs must have a rapid response to complaints about spam sent to abuse@isp.com.
4. ISPs must charge a fee to users who are terminated due to spamming-related activities.

Thomas Gal posted a link (http://www.isp-planet.com/news/2002/vanquish_020906.html) to an old story about anti-spam startup *Vanquish*, whose idea had been that if any email recipient could impose a small fine (say 5c) on any email sender, then such a fine would be a barrier to spammers, but not to legitimate correspondents.

Tim Bedding was interested in the idea, but said that any solution that relies on the market too much "smacks of requiring people to hire security guards when they go for walks, rather than making murder illegal."

Pete McNeil suggested that the 'polluter-pays' model is a mistake, "because it implies that there is some amount of money that will legitimize any content … [the] bad guys can buy a licence to pollute." der Mouse, on the other hand, said that he saw nothing wrong with the idea that bad guys can buy a licence to pollute – providing the money goes to the right place (the victims as opposed to the spammers' ISP/NSP).

Phillip Hallam-Baker asked whether anyone had access to a graph of false positives vs false negatives with the various blacklists plotted on it so that a useful comparison could be made. In response, Pete McNeil posted a link to analysis carried out by himself and Markus Gufler: http://www2.spamchk.com/public.html.

Finally, Alan DeKok observed this month, "Studies have shown that people deleting spam from their inboxes by hand are only 98–99 per cent correct … So for them, losing ~1 per cent of real email is acceptable." This fact had given Alan cause to wonder about the need for a 'perfect' anti-spam system, and he concluded: "The reach for perfection is a noble goal, but not a practical one."

[*As always, an archive of all ASRG postings can be found at http://www1.ietf.org/mail-archive/web/asrg/current/.*]