FEATURE SERIES

The Usual Suspects – Part 3

Andreas Marx
University of Magdeburg, Germany

This concludes our look at the commonly encountered problems thrown up when testing anti-virus software.

Password-protected Office documents

Microsoft Office documents can be password-protected. For *Office 95*, the content of the document and all *Word 95* macros are stored in encrypted form (*Excel* macros are not). However, this encryption is easy to break and it is no big deal to find out if a macro virus is inside it. The encryption has changed in higher *Office* versions and cannot be broken in real-time any more, but the macro modules are no longer encrypted and can be scanned easily.

If the AV program is capable of scaning inside password-protected files like .DOC, .XLS and .PPT, all well and good, but some cannot or do not want to break *Office 95* encryption. If this is the case, a warning message (to the effect that the file is password-protected and cannot be scanned) should be displayed and put in the report file.

Run-time Compressed Files

Script kiddies tend to compress well-known backdoors, worms and other malware to avoid easy detection by the scanners. Under DOS it was relatively easy to decompress the files in memory and check if they looked dangerous. However, in these Win32 times it has all changed – compression, encryption and other protection programs are common and the files can be very large.

It usually takes considerable amounts of time and memory to decompress such files, not counting the time taken to develop a solution for a single compression method of a special version. We are still playing the 'we can't win' battle against viruses; every new virus will get its own signature, so why can't we develop decompression routines for the most common Win32 compressors too? Only a few programs are able to scan such files – this has not changed for a long time now, while the problem gets more significant every day. Another idea would be to compress any known Win32 malware using all known packers and include them into the virus database. This is a very complex process and may be impossible if there are too many different variations of compression programs and malware.

Scan Time and Speed

For a long time, it was thought that only the scan time of an uninfected PC was important, since this would be the standard scenario for an AV scanner. The situation has

changed now, especially on a mail server – since everyone can send infected emails over it, it is important to scan the files fast enough to find viruses. Moreover, cleaning is usually enabled which takes additional time until the email can be delivered.

There are some ItW viruses around which require significant scan time – Win32/Fono is a good example. Large and complex documents with fragmented OLE2 structures need longer still. The interesting question is what happens if several of these files reach the server at the same time? We tried it – some programs experienced a slow-down, but about half crashed after about 30–50 infected files, which would constitute a nice DoS (Denial of Service) attack. With cleaning enabled, it only took half that number.

However, our *Exchange* and *Notes* test system – a *Pentium III 800* with 384 MB RAM – did not slow down. One idea would be to improve the scan time for problem files – not possible without major changes. A more suitable solution would be to implement a better handling of 'in-progress' emails which are currently being scanned or are next in the queue. It also makes sense for customers to switch off disinfection and only quarantine attachments, since disinfection causes its own kind of trouble (see below).

Updates

It is a fact that users want to have an easy Internet download function for updates. However, most still want to be able to download updates separately, when necessary. An offer of regular program updates and upgrades on CD is a good idea.

If a user buys an AV program in a shop, it is often weeks or months old and has to be updated first. Less understandable is if someone downloads a program from the Web and this version is very old too. Why not keep it up to date?

Internet updates seem to be easy – one button has to be pressed and the download starts. However, in our tests, at least one program consistently forgot the proxy settings and in some, neither name nor password for the proxy server was specified. Another problem is caused by programs which always download all files first – after they have finished it looks like something completely new. Another program always downloads the complete databases, after it has checked if they are newer than the last version. Both cause the 'high network traffic' problem for both retail and corporate users. This could easily be avoided.

Newer implementations only download the changed part of files, which requires some extra programming but the downloads are much smaller and faster. After the download, all the changed files are patched and they look like they would after a complete download. One solution would be cumulative and regular small updates loaded directly by the program. However, this requires more memory at run-time and after a while a big new update will have to be released.

Often, like with outbreaks or other dangerous situations, it is useful to be able to have very small update files. In the best case these would be readable ASCII files, which can either be downloaded or sent by email or maybe even fax. Since they are very small, clients can be updated very fast without high network usage and even in an outbreak situation the vendor's Web server may still be fast enough.

One thing we have seen several times is the possibility of updating the engine and the program separately. However, the program can usually only update the signature files, but the engine has to be downloaded and installed separately. In the real-world we foresee a big problem here: many people only download new signatures, not engine updates.

Since AV developers do not test all the possible variations of signature and engine versions (a nearly impossible task), strange things can happen after an update – like system hang-ups or the overwriting of system areas. A better idea would be simply to implement a feature to download engine updates as well as the usual pattern updates, if a new engine is available. After all, the engine update has to implement new detection routines for new sorts of malware which cannot be found with the old engine. The user can have the latest signature file and the program can say it is up to date, but the scanner will only detect a subset of new viruses.

Banana Software

Some updates look like 'banana software', which matures when it is with the customer. It is understandable that not every last detail can be tested on every platform with every software release. However, we cannot see why there have been – for example – so many false positives every now and then on standard applications like DOS 6.22, *Office 97*/*2000* and *Microsoft's* Java implementation.

It is a fact that proper testing requires a lot of time and it can only show if there is an error, not if there isn't. Since updates are released daily or weekly, it is important to test everything sufficiently speedily. Some say that a slight change in the signature file cannot cause problems, but we have seen some, such as the misidentification of viruses which causes a wrong repair routine to be called, or buffer overflows etc. Even minor changes can have significant effects. Another example was the boot-up scanner of one program which ceased to work after a signature update as there was no longer sufficient memory available.

It is a good thing that anti-virus tests can be performed on a parallel basis – the more PCs and humans, the faster the tests. These do not include detection tests, where scanners have to find at least the number of viruses from the last update and then the newly implemented detection, and only then perform disinfection. As mentioned above, no code change does not mean no problem with disinfection – the disinfected files still have to be checked to ensure that they are the same as in the last disinfection tests. Stability (some corrupted files, as well as other kinds of data trash and very large files should be tested), speed (the decrease caused by the new signatures should be minimal and unnoticeable) and all the other software quality criteria have to be tested.

Boot viruses on floppies and hard disks should not be forgotten. Our last test showed the lack of detection of boot viruses in a few products. One program was able to handle Unicode directories and files correctly, but the next version of it was not. It also seems that some developers only test their archive detection against the *EICAR* test file and not against large real-world archives.

It should be noted that we recorded minor differences on the different *Windows* platforms (*98*, *NT*, *2000*) in our last batch of tests and some dramatic differences on *NetWare* (*4.11*, *4.20*, *5.00*, *5.10*) which are all reproducible. The same problem happened with the English but not the German version of *NetWare*.

Memory Detection and Disinfection

A big problem seems to be the detection and disinfection of active malware. Most scanners show that they scan the memory, when most only look for boot and old DOS file viruses in the first 640 KB of it. If so, most of today's viruses will be missed, since they have been written for Win32. If such a virus is active in memory and the user scans all files, it has a good chance of infecting everything.

However, some scanners do this for Win32/CIH – they scan everything, infect everything and clean everything, leaving the infection marker 'U' before the PE header starts (which is not a bad idea). This cleans the system completely but it is not an ideal solution. A better one would be to detect at least all the top twenty non-macro ItW malware in memory and clean them. For viruses this is tricky, but worms can usually be removed using some simple *Windows* functions. Some special disinfection programs and developer freebies can do this easily, but usually the main virus scanner cannot, which begs the question 'why not?'.

Some special malware, such as the Win32/PrettyPark worm or the Win32/SubSeven backdoor, uses a tricky way to hinder an easy disinfection. In the Registry, they change the entry which governs how to start EXE files so that the virus will be started first, which will run the requested executable file. There could be a problem caused by an active resident virus protection – it denies access to a detected malicious program, no other programs can be started any more, and in most products the resident protection cannot be switched off in such a situation. Under DOS, worm parts can only be deleted, and after such a 'repair' programs will not run since the Registry is still unfixed. Only one program avoided this problem, replacing the virus parts with a program that starts the EXE files the usual way and the system still runs.

Back to *Windows* – the problem is first that the Registry fix (including a possible Autostart entry) has to be made and second that the malware files have to be removed, but they are usually active and cannot be deleted. Only one program was able to do it this way – most removed the malware parts but then the system stopped running.

Another example of tricky disinfection would be that of Win32/Ska (aka Happy99) which changes WSOCK32.DLL. While disinfecting, the file has to be replaced by a backup copy from the installation CD or by a backup copy the worm creates during infection. However, only a few programs do this – most delete WSOCK32.DLL or leave it in a modified state. A well-known ItW worm with backdoor functions is Win32/QAZ. With this worm, the file called NOTE.COM must be renamed NOTEPAD.EXE (after deleting the viral NOTEPAD.EXE) but only one program did this in our last test. Only two programs were able to remove the Autostart entry – out of 13 scanners tested. AV companies should not document large disinfection instructions with up to 20 difficult steps if they can be automated.

Of course, users should be warned in the case of a critical disinfection and also if a virus like Ripper or XM/Compat has infected the system, since these two viruses change the user's data randomly. Win32/MsInit (aka RC5) causes a problem, too – one part of it is a harmless program, but the user should know about its installation and resource consumption. Maybe a macro virus warning feature can be restored in the Registry too, or the program can display a warning about it.

File and Macro Disinfection

Disinfection of DOS viruses did not often result in problems – sometimes files were simply a little bit longer, since the virus did not store the original length. With Win32 it is a very complex task to restore everything so that the program is still able to run. This can easily be seen by comparing the disinfected files of different programs – usually no two repairs are identical (for DOS, it is). This also causes the problem of a scanner creating a new variant of an existing backdoor or worm program when it cleans it the virus incorrectly and does not check for further infections or cannot find the wrong, cleaned malware program any more. There are many examples of this, such as a Win32/CIH-infected Win32/Back_Orifice backdoor.

Macro disinfection is a complex task too, especially if the program tries to keep the user macros intact. More than half the programs we tested had problems with this, however the results looked better after every test. Most are just inconvenient, such as error or macro virus warning messages while opening a disinfected *Office* document or while opening the VBA Editor even if no macro is inside the file. In some cases this is easy to avoid – *Office 95* files can be cleaned very easily so that neither *Office 97* or *2000* displays a virus warning message.

It is more difficult to clean *Office 97* or *2000* files correctly and after we compared disinfection methods we can say that every company's methods differ slightly, with better or poorer results. There were only two programs which cleaned the files so 'well' that they could only be opened with many error messages on an English version of *Office*, and not at all on a localized German version.

We keep testing all possibilities of parasitic and non-parasitic DOC and XLS infections. The first problem occurs while scanning parasitic infections: some programs are unable to detect the virus any more, even if the program is still working fine. The second problem is the disinfection: sometimes user macros are removed (in the case of parasitic macro viruses this is OK), sometimes the ThisDocument stream is destroyed, sometimes the modules are corrupted and sometimes the file or the VBA editor cannot be opened nor could macros be created or started. More than one program had a nice idea for disinfection: all macros (sometimes including the user macros) were shortened – only the 'Sub <Name>', some spaces and 'End sub' were left – with interesting results, if the virus uses functions like FileSaveAs…

Conclusion

This series has covered only a number of important issues concerning the oft-encountered problems of anti-virus programs. There are a lot of others, such as a good central administration program, but much has been written on this. There are still many improvements needed for groupware anti-virus software, for example, not only allowing black-listing, but also white-listing of file types using smart scanning. It should also be noted that anti-virus programs cannot constitute the future in our connected world, but together with other kinds of software, like desktop firewalls and content filtering programs they may help to make the problem easier to handle.

I'd like to thank the people who gave me comments and suggestions for this articles and our tests, especially Sarah Gordon, Eugene Kaspersky, Igor Muttik, Petr Odehnal and Costin Raiu. Readers are asked to refer to our exhaustive comments on disinfection results for current tests on our newly-designed Web page http://www.av-test.org.