FEATURE SERIES

The Usual Suspects – Part 2

Andreas Marx
University of Magdeburg, Germany

We continue our look at the problems commonly encountered during the testing of anti-virus products.

Language

For server or mail server systems an English version of the AV scanner is sufficient. However, client systems require a localised version of the program, since not everyone speaks English. The program's implementation method is often to recompile itself with new language strings. This is not a good solution since even minor changes or patches require programmers to recompile everything in several different languages before testing can be carried out.

A better idea would be to provide localisation files for the scanner and make the main program independent from the language. This can be done using language definition files where all the important strings are stored, or DLLs with resource information for *Windows* platforms which are easier to handle.

Bootable Start Disks/CDs

It would be neither expensive nor labour intensive to provide customers with slightly better protection in emergency situations by making the installation CD bootable. Most programs use a small *Linux* kernel and a *Linux* version of the scanner to scan FAT, FAT32 and NTFS volumes and this is completely free of charge for the company. DOS and a CD-Rom driver will work for FAT and FAT32 disks too, but usually licence fees have to be paid. Some companies avoid this by writing their own simple DOS with additional routines to avoid problems with special malware (mostly tricky boot viruses).

Since there are still many old computers around, a bootable floppy disk should still be included in retail products. However, it does not make sense to include up to seven disks with the main scanner program running under DOS. It would be better if the bootable disk worked like the bootable CD and loaded the main program from the CD and additional or updated databases from floppy or hard disks.

Virus Naming Conventions

Very often, different products have different names for one and the same type of virus. This starts with a prefix like Macro.Word97, W97M, WM97, followed by '.', '/', '_' or whatever. Some programs use strings like O97M to show that a virus is able to infect more than one *Office* platform, others use the optional @mm or @m to show that it is a mass-mailer or a mailer. This is where it stops being relatively easy. For Win32 file viruses and worms there are more than 30 different philosophies and suffixes (Win32, Win95, W95, PE, I-Worm, TROJ etc.) and we need a standard supported by the majority of companies.

The same confusion surrounds virus names – the most widespread malware should have one and the same name under all scanners. In emergency situations different names are understandable, but never changing the name after including signatures into the database causes confusion. Another problem is the variant detection of some programs. Some say they have definitely found 12345.A, but it is actually a completely different variant since the identification checks just a few bytes. In this case, a less precise name would show that more than just one variant could be identified and that this identification is generic.

Self-Checks

Every security software product should perform a self-check to make sure it is in the original, unmodified state. However, some AV programs do not perform a self-check at all, neither on the main program, nor the scanner libraries or virus databases. Installation need not be checked, but these three essential parts ought to be.

We have seen several methods of integrity check: starting from an easy 8-byte XOR through a CRC16 or CRC32 up to a strong cryptographic checksum. The last is probably the best idea, especially if the databases contain executable code or p-code, which allows write-access even in 'scan only' mode and not just for disinfection or archive handling. In the past there have been some retro viruses which successfully caused problems with deletion or modification of all scanned files.

The check must be performed before a value of the bases is read for use in the scanner engine, since a wrong value could cause buffer overflows or crashes. If the program or the databases have been modified, an error dialog must be displayed containing all the information needed to clarify the problem and instructing the user on what to do.

This includes messages to the effect that the program must be re-installed or a scan performed after booting from a clean disk. Sometimes only short dialogs like 'ScanInit failed.' or 'Error 128. Reinstall product.' are displayed. This is not good enough. Other programs, while they do not load virus databases if they are corrupt, do not display suitable warning messages.

If the scanner seems really fast it may be because there are no viruses for it to scan for or only a very few. In one case we saw the scanner really slow down because the heuristics had to do everything. If the program is rather old, an

appropriate message should appear advising that the scanner should be updated as soon as possible and maybe how to do it. With server or mail server systems this can be done via email or in other ways.

Encryption

A good encryption of all virus-related parts in the program and its databases helps avoid false positives from other scanners and reverse engineering by virus writers. Since the size of the databases increases fast, they should be compressed, too. It is incomprehensible why some leading companies still only use '+1' or 'XOR 255' encryption of their work – some competitors' X-Ray engines are able to look inside these files! On the other hand, it makes no sense to implement strong AES (Advanced Encryption Standard) routines for protection, since the scanner has to be able to read everything. If someone really wants to decrypt the databases, they will succeed eventually.

In the main program or scanner libraries there are often unencrypted heuristic strings for macro or script virus detection or proper removal, such as modified Registry keys and how to restore them – these should be secured.

On-Access Scanners

A virus guard has to protect its user against the same viruses which the on-demand scanner finds. However, some programs do not allow the specification that all files should be scanned and that not only incoming (writes) but also outgoing (reads) files should be scanned, too.

Under high workloads it is possible to spot really big problems. In these situations, some scanners are unable to scan all files or crash intermittently and cease scanning altogether. The same thing tends to happen to the program displaying alerts and writing them to a log file: after several infections it either crashes or fails to display a full list of all the viruses found.

In some cases, it can be useful to switch off the guard for a period of time, for example while burning a CD. What we do not understand is why some scanners require a *Windows* restart for small configuration changes or after unloading. The virus guard can also help protect the scanner against modifications by retro viruses or Trojans, since it looks at access to all files. So, it is easy to implement a routine that checks if a program wants to modify or delete one of the scanner's program files and avoid it.

Archive Formats

A good scanner should be able to detect viruses in popular archive formats, and at least in ZIP files. A survey for the preparation of our last test showed that customers also want ARJ, CAB, LHA, RAR and ACE-compressed files, as well as Unix formats like TAR, GZ and BZ included in the list. Since TAR files are not compressed, some scanners randomly detect viruses inside this type of file.

Of course, the scanner should be able to scan inside the files recursively (ZIP in ZIP, but also GZ in TAR archives) in both GUI and command-line versions (DOS32 and higher). In some cases, people answering the survey requested that it should be possible to include external unpacker programs if a file format (e.g. ACE) is not supported by the scanner.

It is odd that, even if some scanners can handle archives correctly, the same programs may be unable to scan inside self-extracting (SFX) files of the same type, since the same decompression routines can be used. Other scanners only look for known SFX unpack routines, but will fail on new, changed or different language versions of them. It is essential to support most installation archives (InstallShield, Package for the Web, etc).

While scanning archives in memory is the faster and more secure solution, about half of the scanners we test are unable to do it: they extract the archives into a temporary directory and scan them afterwards. With this method, every file has to be renamed to avoid problems with specially prepared file names including pipes ('|') or other problematic characters like '`' or '". Names of subdirectories have to be ignored, since viruses like BAT/WinRip use '..' constructions to spread and copy themselves into the *Windows* Autostart directory.

Even if a scanner supports many file types, a useful standard setting – for example, 'scan only 5 recursion layers deep' – is important, since unpacking requires a lot of memory and stack space. Very large files can cause problems, too – some scanners will skip them without any notice, while others require time to scan inside them and it looks like the scanner has crashed.

Embedded OLE objects

A good anti-virus scanning engine should be able to scan embedded files inside OLE files numerous times without problems and handle them like an archive file. However, some programs still fail to find an infected .DOC in an XLS or SHS file.

In our tests, we only look at the most significant scenarios – infected COM, EXE, VBS, DOC, XLS and PPT files embedded into DOC, XLS, PPT, SHS and even RTF files. Usually, only about half of these will be found, and RTF files will not be scanned at all. But there are additional formats to these, since *Office 2000* supports the saving of all documents as HTML files, storing macros inside OLEDATA.MSO or EDITDATA.MSO files. Such files should be scanned, too, regardless of whether they include either additional embedded objects or the original document was infected.

Next month's final instalment of this series will focus on the following issues: password-protected *Office* documents, run-time compressed files, disinfection, speed, updates and test strategies.