The Use and Misuse of Test Files in Anti-Malware Testing

Notice and Disclaimer of Liability Concerning the Use of AMTSO Documents

This document is published with the understanding that AMTSO members are supplying this information for general educational purposes only. No professional engineering or any other professional services or advice is being offered hereby. Therefore, you must use your own skill and judgment when reviewing this document and not solely rely on the information provided herein.

AMTSO believes that the information in this document is accurate as of the date of publication although it has not verified its accuracy or determined if there are any errors. Further, such information is subject to change without notice and AMTSO is under no obligation to provide any updates or corrections.

You understand and agree that this document is provided to you exclusively on an as-is basis without any representations or warranties of any kind whether express, implied or statutory. Without limiting the foregoing, AMTSO expressly disclaims all warranties of merchantability, non-infringement, continuous operation, completeness, quality, accuracy and fitness for a particular purpose.

In no event shall AMTSO be liable for any damages or losses of any kind (including, without limitation, any lost profits, lost data or business interruption) arising directly or indirectly out of any use of this document including, without limitation, any direct, indirect, special, incidental, consequential, exemplary and punitive damages regardless of whether any person or entity was advised of the possibility of such damages.

This document is protected by AMTSO's intellectual property rights and may be additionally protected by the intellectual property rights of others.

The Use and Misuse of Test Files in Anti-Malware Testing

Introduction

Traditionally, when asked for samples with which to "test" an anti-virus program, anti-malware researchers have advocated [1] the use of alternatives to the sharing of truly malicious samples to anyone who doesn't have access to malware through mainstream, trusted channels, as a way of simulating malware behaviour without the attendant risks of using genuinely malicious programs [2]. The most commonly suggested and used alternative to real malware is the EICAR test file, a utility intended for checking that security software has been installed and is active, named after and under the sponsorship of EICAR. But the EICAR file is not and cannot be suitable for the entire range of scenarios in which we sometimes see it used [3].

While the creation of *simulated* malware may seem less contentious than the creation of actual malware [4], generation or modification of otherwise non-malicious test files may present more problems than the aspiring tester may realize. This document derives from the realization that there has been increased and unanticipated use of the EICAR file in testing contexts for which it was not designed - or appropriate. While it might be regarded as a classic survivor of the strictest form of signature detection, its use even in static testing is very limited indeed. Furthermore, while almost all traditional antivirus programs detect the EICAR file, that detection is neither mandatory nor universal.

While this paper's main focus is on the much misused EICAR test file, discussing the separation in functionality between its use as an installation check and its use as a limited test tool, it also looks at some other test files sometimes used in comparative testing.

Virus Simulators

Sarah Gordon [5] classified simulators as follows:

- **Simulators for Education** – demonstration programs such as the Virus Simulation Suite, Virlab, and AVP. These, she suggested, raise awareness, but mislead by setting up false expectations as to behaviour. This observation seems more accurate than ever in an era where most malware, being intended to make profit rather than a visual demonstration of some amateur coder's prowess, runs as inconspicuously as possible to reduce the likelihood of detection and removal.
- **Tests Using Simulators** – in particular, Rosenthal's Virus Simulator. While the simulator marketed for many years by Doren Rosenthal is rarely, if ever, seen in contemporary tests, it remains a prime example of a testing model that is both ethically and technically flawed [6].

- It was based on the false premise that a product that detects a real virus should also detect a simulation of that virus. [7]
- That premise reinforces the equally false premise that a virus signature is some unique footprint and that all security software uses the same signature. [6, 7] There is, of course, no reason why random virus fragments should be detected as if they were a real virus, and even less likelihood that several vendors will use the same signature (in the broadest sense of the term).
- The registered version of the utilities also included a real – if relatively innocuous – virus [4].
- The inappropriate use of Rosenthal's programs by some testers in comparative testing [5] forced the AV industry to add detection of yet another non-virus to the impressive selection of garbage files, "intendeds" (objects *intended* to be malicious that cannot, however, execute and therefore fail to fulfil that intention) and other inappropriate objects it needs to detect if a product isn't to be penalized for *not* detecting what it *shouldn't need* to detect [8].

Even if we ignore the ethical issues [4] regarded by some critics of the antivirus industry (and, by association, of the Anti-Malware Testing Standards Organization) as a purely self-serving/self-protective objection [9], the technical objections raised by both Gordon [5] and Sambucci [10] in the 1990s are no less true today.

The purpose of the EICAR Test File, as summarized by Gordon [5] and others [2], is to establish:

- Whether AV is installed "correctly"
- What happens when the AV finds a virus
- Which messages are displayed
- How it handles "custom warnings", batch files, and notifications to the system administrator over the network.

Gordon states that "the existence of the EICAR test file, we feel, obviates the need for simulated viruses used for testing purposes". She was writing at a time (1995) when the malware threatscape was largely dominated by a few comparatively slow –spreading boot sector viruses and classic file infectors like Jerusalem [11]. There is even less use or need for simulated malware (or non-simulated custom-generated malware, for that matter [9]) in an age of malware glut, when virus labs are seeing tens and even hundreds of thousands of unique malicious binaries per day. However, it seems that expectations of what the EICAR file can reasonably be used for have been set too high. [1]

The EICAR Test File

The EICAR test file was never specifically or overtly intended as a tool for testing in the sense of either comparative detection testing or testing for certification purposes [12], though correct detection of a de facto industry-standard test file is certainly a legitimate (if *very* limited) test target. Apart from that, it's

hard to see any useful purpose for it in a test intended to compare or evaluate detection testing, except to confirm that the scanner is active, as described below.

Rather, it's intended as an installation check (or test). Perhaps the use of the term "EICAR test file" is unfortunate in that "by default", people now think of testing in terms of comparative testing or certification. However, getting the world to refer to it as the "EICAR installation check file" is "probably as hopeless as restoring the original and non-pejorative use of the term hacking" [1].

When the EICAR file became almost universally supported for installation checking by mainstream antivirus companies, it replaced not only the widely-deprecated Rosenthal simulator, but also files with similar functionality created by individual vendors for use with their own products. Clearly, it was preferable to use a test object detected by all mainstream vendors in approximately the same way across platforms.

Technically, however, it lacks all the characteristics that we normally associate with self-replicating malware. Most obviously, it doesn't replicate, though it can be and has been described as simulating an overwriting virus [10]. It has no malicious payload, and can't even be described accurately as a Trojan horse, since all it does when executed is display a message identifying itself. In other words, its functionality is entirely dependent on the precise identification of the byte sequence of which it consists. Namely, the following string of characters – actually, the specification allows for a (very little) bit more than that, but we'll come to that.

```
X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*
```

If allowed to execute (though an active on-access antivirus scanner *should* normally prevent it from executing), it prints the following substring to the screen.

```
EICAR-STANDARD-ANTIVIRUS-TEST-FILE!
```

Note that while detection and flagging of the EICAR test file during both on-demand and on-access scans can be described as a de facto industry standard, it isn't mandatory for anti-virus to conform to that standard, so recognition or non-recognition of the EICAR string is not, strictly speaking, a suitable criterion for evaluating its effectiveness.

The EICAR file doesn't tell you much apart from saying what it *is*, of course: its only function is to check that a conformant anti-malware program is installed and "awake". It doesn't, in itself, prove that AV is configured properly. It doesn't even prove it detects any real viruses. [2]

Unlike most of the other "simulations" listed by Gordon, the EICAR file was never intended to be a realistic malware simulation. In fact, it's a matter for debate whether simulated malware can ever behave like "real" malware in any sense that makes it useful in detection testing without actually being malware. However, as long as the EICAR file meets EICAR's own specification correctly [1], it exhibits no malicious behaviour of its own. It simply aims to generate a response from security software that approximates to the response that real malware would generate.

While most mainstream products will identify it, and usually won't allow it to execute (often quarantining it or limiting access in other ways), the way in which a scanner responds is rarely exactly the same as that elicited by real malware. Even displayed messages are likely to differ from standard messages, reflecting the fact that most security researchers consider it inappropriate to flag a non-malicious file as malicious. Hence such messages as "EICAR test file not-a-virus detected". (That's not a real example, but it's not far-fetched either.)

Strings Attached

The EICAR test file is atypical of real malware in another way. Whereas the anti-malware industry has become highly focused on generic and proactive technologies (under the various but related umbrellas of heuristic analysis, behaviour analysis, sandboxing, emulation and so on), the EICAR test file is a classic survivor of the strictest form of signature detection, sometimes referred to as exact identification.

Exact Identification can be defined as the recognition of a virus when every section of the nonmodifiable parts of the virus body is uniquely identified: in principle, the same applies to non-viral malware. In this case, identification is even more atypical in that:

1. The entire "sample" is an ASCII text string, even though (apart from the substring which is actually displayed on execution) it's a rather goofy-looking string. This is deliberate: the file was always intended to be strictly ASCII characters so that it could be created by typing the string into a text file using a simple text editor. [12]
2. More crucially, the entire "sample" (appended whitespace characters apart) *is* the nonmodifiable code.

To understand exactly why this is important, we first have to know a little of the history of the EICAR file.

Evolution of the EICAR File

The EICAR test file was created [12] by members of CARO for EICAR in the early 1990s. CARO (Computer AntiVirus Researcher's Organization) is an informal group of individuals who have worked together since around 1990 across corporate and academic borders to study the whole range of computer malware. [2, 13] Whereas CARO was always a technical group, EICAR also had a distinct legal and general security focus. Nowadays, the two groups operate largely independently of each other, but there is of course considerable overlap in membership and objectives.

The EICAR test file was a historic joint project, created by CARO members and published by EICAR, in order to meet a perceived need for a simple means by which a helpdesk operator could read it over the telephone to an end user, to allow a check on whether his or her antivirus was working. [12] The original definition, in short, was as follows [2]:

> "The file is a legitimate DOS program, and produces sensible results when run (it prints the message "EICAR-STANDARD-ANTIVIRUS-TEST-FILE").

> It is also short and simple - in fact it consists entirely of printable ASCII characters, so that it can easily be created with a regular text editor. Any anti-virus product that supports the test file should detect it in any file providing that the file starts with the following 68 characters:
>
> X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*
>
> To keep things simple, the file uses only upper case letters, digits and punctuation marks, and does not include spaces. The only thing to watch out for when typing in the test file is that the third character is the capital letter "O", not the digit zero."

However, events during the years between the original specification and 2003 necessitated a rethink. A virus commonly known as Bat/Bwg.a@MM made use of the EICAR file as a stealthed approach to infection. Bat/Bwg.a@MM starts with the EICAR string: when the worm is run, it generates a "File not found" error but the execution continues. Many AV products incorrectly detected this malware as the EICAR test file when it first appeared. While this was the first widelyknown instance of malware impersonating the EICAR file, it wasn't actually the first problem deriving from a slight looseness in this initial definition [6]. While the definition is entirely accurate as a definition of what the file actually is (or was at that time), it assumed a commonsense approach on the part of all AV vendors and therefore didn't go into detail on what the file *could not* or *should not* be. This led to such anomalies as detection of files being detected as the EICAR file because they contained the EICAR string, even where it didn't constitute the very first characters of the file.

EICAR therefore wanted to help the AV industry and its customers by introducing a slight change to the formal definition so that a fully-regularized, correct, safe definition was available that would leave no doubt in the minds of either vendors or users as to what a fully conformant EICAR test file should look like.

The definition agreed between EICAR and the AV industry [14] now looks like this:

> "The file is a legitimate DOS program, and produces sensible results when run (it prints the message "EICAR-STANDARD-ANTIVIRUS-TEST-FILE").
>
> It is also short and simple - in fact, it consists entirely of printable ASCII characters, so that it can easily be created with a regular text editor. Any anti-virus product that supports the test file should detect it in any file providing that the file starts with the following 68 characters:
>
> X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*
>
> The first 68 characters is the known string. It may be optionally appended by any combination of whitespace characters with the total file length not exceeding 128 characters. The only whitespace characters allowed are the space character, tab, LF, CR, CTRL-Z."

Thus, while the file was always essentially a string of 68 characters that have not changed between definitions, the enhanced definition, as long as it is scrupulously followed, comes close to eradicating

risk of false negatives (malware managing to execute because its presence is masked by the presence of the test string). It also lessens the risk of a special case of false positive: that is, when the EICAR file is detected when it shouldn't be (for example, when embedded in a document as above) [6]. White space apart, the file was always intended to be an exact sequence of bytes.

A number of ingenious ways have been used to extend the functionality of the EICAR file, some of them genuinely useful [15], but some misleading and inappropriate because they were based on a misunderstanding of the specification, or ignored it completely [16].

Mainstream testers have privileged access to sample repositories and sample exchange as a result of validated, trustworthy contact with each other and the anti-virus industry, including contractual agreements and joint membership of forums and groups such as AMTSO

Attractive though it is for aspiring testers *without* those contacts to be able to test safely [17], mixing up EICAR detection with malware detection creates more potential problems than it solves. In extreme cases, we've seen instances [18] where a single test has attempted to assess:

- Recognition of the EICAR test file (not necessarily an invalid objective, but that depends on what *conclusions* are drawn).
- Recognition of presumed (but unvalidated) In the Wild (ItW) malware.
- Recognition of presumed (unvalidated) malware not known to be ItW
- Recognition of presumed (unvalidated) malware not expected to be known to the scanner

At the very least this indicates a muddled and, therefore, undependable methodology.

However, there have been a number of attempts to assess and compare the effectiveness of multiple antivirus products using some form of modification of the test file itself. In a series of crosspostings to alt.comp.virus, bugtraq et al [16], "keepitsecret" suggested that the test file could be used to "learn how AVs do their job...watch how heuristics work with a code in principle detected by its signature (somehow, a way to assess the limits of this method)..."

This, like a number of similar attempts to draw conclusions about how AV products work, was doomed to failure for two reasons.

Firstly, it failed to take into account the narrowness of the specification. In general, six different scanners may identify a known malicious program equally effectively, yet get there by very different routes. EICAR's strict definition, however, introduces limitations. Whatever the mechanism, the underlying algorithm is still along the lines of:

IF

{file starts with the unmodified 68-byte EICAR string}

AND

{file length is less than or equal to 128 bytes}

AND

{appended characters are all included in the set [space, tab, LF, CR, ^Z}

THEN

ECHO “This is the EICAR test file”

Secondly, the strictness of the definition relieves the developer of any mandate to do anything with an object that doesn’t meet the specification *unless* it happens to set off a different detection algorithm that identifies it as malware (or “might be” malware, or “possibly unwanted”, or some other label that brings it back into the range of programs that should be detected). After all, it doesn’t do anything malicious, so there’s no reason to detect it when the displayed character string is no longer the string defined in the specification.

Clearly, a version that displays "EICAR-STANDING-ANTIVIRUS-TEST-FILE!" isn’t the test file, isn’t malicious, and needn’t be detected. Some vendors have chosen, at various times, to display a message such as “EICAR_Test (modified)”, but it’s debatable whether this is a good idea, since it causes confusion. It may or may not be “appropriate” detection behaviour, but that doesn’t make ignoring a modified version of the EICAR file “inappropriate”. (Detection of or failure to detect a maliciously modified version is a different issue.) Identifying a modified EICAR display string as the EICAR file is just plain wrong.

Other modifications that have been suggested in the past include:

- Prepending a NOP or JMP
- Using XOR and OR encryption, or other cryptographic techniques
- Using polymorphic code
- Adding replicative or file-searching code
- Splitting the file
- Padding with or appending additional characters

While in general products will disregard trivially modified versions, as they bear less resemblance to EICAR and more to a possibly malicious program, more scanners will start to detect them generically. However, such changes constitute a seriously unreliable guide to detection performance. Even worse, changes that “resemble” a malicious program are of no value for testing purposes unless they really are in some meaningful sense malicious, but creation of malicious programs (irrespective of any relationship to the EICAR file) is subject to a different and very significant set of problems. [4]

CloudCar

CloudCar can be defined in short as a test file that may be provided by any vendor that provides a cloud solution.

Many security vendors are enabling cloud based solutions. These solutions rely on specific and reliable network communication for proper functioning. Testing labs construct elaborate systems to simulate the real world in order to conduct their tests and monitor results safely. It is possible that while constructing these systems the network rules imposed may interfere with a product's ability to communicate correctly with its backend servers. This is where CloudCar comes in. CloudCar is a file that will only be detected by a product's cloud service. Scanning CloudCar with the network disconnected (or inoperable) must not result in a detection. Scanning with the product connected properly to the internet must result in a detection. With CloudCar a tester can verify that a setup for a given product allows it to communicate properly with its cloud service.

Each vendor will have its own CloudCar and these files should never have a traditional detection written for them because they are not malicious. The CloudCar executable itself need have no specific functionality. It simply must have a unique hash.

Spycar

Spycar (http://www.spycar.org/) came out of a research project at Intelguardians Labs, as a result of collaboration between Ed Skoudis, Tom Liston and Mike Poor. It was intended to test anti-spyware programs by observing their response to certain behaviours commonly associated with Windows spyware, using tools that were not malicious. Like Rosenthal's virus, they made no permanent changes to the system. Not everyone was convinced by the methodology, though the idea of checking behavioural detection rather than signature detection was by no means totally invalid. However, while the self-conscious "homage" to the EICAR name might lead you to expect a kind of EICAR file for spyware, the Spycar team nailed both the difference between the intent behind the two approaches and the functionality of the EICAR file rather well.

> "The EICAR file can be used to verify that your anti-virus tool is alive and running. Spycar tests behavior-based alerting and blocking. ... You've got a smoke detector, and you want to see if it is working. The EICAR file is like the big red test button on the smoke detector ... the smoke detector beeps, telling you that the battery is charged and everything seems to be working properly ... Spycar... mimics the behavior of a real fire (again, in a benign fashion) to see if your smoke detector is protecting you."

Well, we might question "everything seems to be working properly". To quote the alt.comp.virus FAQ [2]:

> "It's a (limited) check on whether the program is installed, but I'm not sure it's a measure of whether it's installed correctly. For instance, the fact that a scanner reports correctly that a file called EICAR.COM contains the EICAR string, doesn't tell you whether it will detect macro

viruses, for example. In fact, if I wanted to be really picky, I'd have to say that it doesn't actually tell you anything except that the scanner detects the EICAR string in files with a particular extension. "

Still, the Spycar definitions illustrate quite clearly the difference between an installation check file and an attempt to create a type of tool that could, in principle, be of potential value in evaluating products.

In practice, however, Spycar has long since been of little practical use as an evaluation tool because vendors can (and do!) write detections that are based on static signatures as well as behaviour. [19] While it's probable that all mainstream vendors are capable of detecting the (presumed spywarelike) behaviours Spycar used as a test of their efficiency, they don't have to. In fact, Spycar has joined a long list of tests that attempt to measure effectiveness by criteria without recognizing that mainstream AV uses multiple detection techniques that may be effective in the real world even though they ignore those specific criteria.

Conclusion

The EICAR test file has only the most limited application to and connection with testing as the term is normally understood (i.e. comparative and certification testing) by AMTSO [20], security vendors and testers, and most members of the public, and it would probably be more appropriate and less confusing to refer to it as the EICAR installation check file, or something similar, though this is unlikely to happen.

- It's intended as an installation check, not for detection testing. It doesn't tell you whether it's installed optimally, or how it detects real malware, and has no place in a test intended to evaluate detection of real malware.
- It's useful as an installation check in that most scanners detect it, even on platforms where, as a DOS executable, it can't execute natively (for example under Mac OS where no DOS/Windows emulation is in place). However, the way in which a scanner responds is not standardized: neither the exact detection method, nor the way in which detection is flagged and processed, are laid down in the EICAR specification. It cannot and should not be assumed that the way in which a scanner behaves when it detects the EICAR test file is identical to the way in which it will behave when it detects real malware.
- Because of the very tight specification, modification to the core executable will normally invalidate testing, though the use of some sort of wrapper may be acceptable and even useful (for example, storing it in an archive file [1, 15]), as long as the file itself is not modified. A scanning product may be designed to be flexible enough to recognize the EICAR file when harmless modifications are made, but that's a design decision that isn't really a suitable target for testing, as it's not a significant indicator of effectiveness in detection. A product that fully supports the EICAR specification but is not flexible enough to recognize modifications to the core code is *not* behaving incorrectly, and may be more "correct" than a scanner that is more flexible. A scanner that flags EICAR and EICAR variants differently could be said to be behaving appropriately as long as it doesn't present opportunities for malware to "piggyback" the test file. However, it's hard to see what practical use a test to determine those characteristics would be in the real world.

- It's possible to use the EICAR test file to test characteristics and issues that are related to detection but don't require the use of real malware samples. [25]. However, it would take a great deal of care and experience to use such techniques accurately in the context of a comparative test: a tester who had that skill and experience would probably also have the facilities to enable a more direct approach using real malware.

AMTSO has no wish to disparage a utility that has proved useful in many contexts as an installation check over the years. However, as a tool for comparative testing, given the limitations imposed by its formal definition, use of the EICAR file in its present form is rarely (if ever) appropriate: nor is there any obvious application of the Spycar or CloudCar facilities in detection testing. In fact, CloudCar is a special case of a product-specific utility to test configuration settings. While this application of the "is it active?" principle that underlies the EICAR test file might be usefully extended to other tools testing other functionalities – for instance, whether product-specific detection of "possibly unwanted" programs is enabled – it's unlikely that a single generic binary would be acknowledged and detected by multiple vendors in the same way as the EICAR test file, if only because of the risk that it would be used inappropriately in comparative testing, as has happened with the EICAR file. It's possible that a toolkit of functionality testing programs may have wide application, but its use in testing would be limited at best since changes in technology could render tools obsolete in much the same way that Spycar has been.

It is certainly misleading to describe the EICAR file as simulated malware, at least in its prescribed form [1]. Is there a place at all for "real" simulation in detection testing? Philosophically speaking, there is an obvious contradiction: if an object isn't intended to do harm (depending on what you understand by the term "harm"), can it be described as malicious? If not, should it be detected by security products as if it were? Generally, the answer from the security industry has been an emphatic "no", even where individual companies have taken the pragmatic decision to add a detection that they consider inappropriate rather than be penalized for failing in tests that use it. [7]

Recently, an online banking security test has attracted some media attention by taking a cohort of security products (including but not confined to internet security products with an AV component) and trying them against its own simulator. Of course, the idea of a test that evades the necessity of operating within a real or virtual botnet environment has its attractions. There have also been a number of somewhat similar tests that simulate attacks using exploits to evaluate performance of firewalls and internet security products. However, these simulations are prey to the same generic objections as described in the preceding paragraph, even if the audience is prepared to accept the company's assurances that the simulation is sufficiently similar to real malware to be an appropriate test subject. In the case of simulated malware, it's also a matter of whether it's safe to assume that the number of *instances* of the exploit is statistically adequate. And the question always lingers in the background: is the tester's understanding of the way that a security product works sufficient to ensure that it's reasonable to expect it to detect the simulation, or is it coloured by a mistaken conviction as to how a product "should" work?

Citations

[1] http://eicar.org/anti_virus_test_file.htm

[2] David Harley, alt.comp.virus FAQ, http://www.faqs.org/faqs/computer-virus/alt-faq/part4/

[3] David Harley, Eddy Willems, and Lysa Myers, "Test Files and Product Evaluation: the Case for and against Malware Simulation in Testing

[4] AMTSO, "Issues Involved in the Creation of Samples for Testing" www.amtso.org

[5] Sarah Gordon, "Are Good Virus Simulators Still a Bad Idea?", Network Security, Volume 1996, Issue 9, September 1996, Pages 7-13. Available online at http://www.sciencedirect.com/

[6] David Harley, Robert Slade and Urs Gattiker, "Viruses Revealed", Osborne 2001

[7] Joe Wells et al., Open Letter 2000, Cybersoft Whitepaper

[8] Peter Kosinár, Juraj Malcho, Richard Marko, and David Harley, "Testing Exposed", http://go.eset.com/us/resources/white-papers/Kosinar-etal-VB2010.pdf

[9] David Harley, "l'AMTSO Mysterioso and Malware Creation," AMTSO Member Blog

[10] Luca Sambucci, Virus Simulator Test. Italian Computer Antivirus Research Organization. 1994.

[11] November 1995 WildList, http://www.wildlist.org/WildList/199511.htm

[12] Padgett Peterson, personal communication

[13] Kenneth Bechtel and David Harley in "The AVIEN Malware Defense Guide" (ed. Harley), Syngress 2007

[14] Eddy Willems, "The Winds of Change: Updates to the EICAR Test File", Virus Bulletin June 2003

[15] Randy Abrams, "Giving the EICAR test file some teeth", Virus Bulletin 1999 Conference Proceedings

[16] "Let's Have Fun with EICAR test file": http://archive.cert.unihttp://archive.cert.uni-stuttgart.de/bugtraq/2003/06/msg00251.htmlstuttgart.de/bugtraq/2003/06/msg00251.html; http://marc.info/?l=ntbugtraq&m=105733144930014&w=2]

[17] The Future of AV-Testing: EICAR position paper, http://www.eicar.org/information/infomaterial/eicar_positionpaper_web.pdf [6] David Harley and Andrew Lee, "Call of the WildList: Last Orders for Wildcore-Based Testing?", Virus Bulletin 2010.

[18] David Harley, "Untangling the Wheat from the Chaff in Comparative Anti Virus Reviews", Eset.com Resource Paper

[19] Ed Skoudis, "What is Spycar?", http://searchsecurity.techtarget.com/expert/KnowledgebaseAnswer/0,289625,sid14_gci1286802,00 .html

[20] Anti-Malware Testing Standards Organization, http://www.amtso.org

This document was adopted by AMTSO on February 24, 2012