J Comput Virol (2007) 3:65–74
DOI 10.1007/s11416-007-0041-5

EICAR 2007 BEST ACADEMIC PAPERS

A statistical model for undecidable viral detection

Eric Filiol · Sébastien Josse

Received: 12 January 2007 / Accepted: 9 March 2007 / Published online: 4 April 2007

Abstract This paper presents a statistical model of the malware detection problem. Where Chess and White (An undetectable computer virus. In: Virus Bulletin Conference, 2000) just partially addressed this issue and gave only existence results, we give here constructive results of undetectable malware. We show that any existing detection techniques can be modelled by one or more statistical tests. Consequently, the concepts of false positive and non detection are precisely defined. The concept of test simulability is then presented and enables us to gives constructive results how undetectable malware could be developped by an attacker. Practical applications of this statistical model are proposed. Finally, we give a statistical variant of Cohen's undecidability results of virus detection.

1 Introduction

The most essential result in computer virology is Cohen's undecidability theorem of virus detection [3]. This theorem states that there is no algorithm that can perfectly detect all possible viruses. In 2001, Chess and White [8] pointed out that there are computer viruses which no algorithm can detect, even under a looser definition of detection, thus extending Cohen's result. To be more precise, suppose we get a detection algorithm D for a given virus v. We can forgive this detector for claiming to find v in some program P which is not infected with v, provided that P is infected with *some* virus. Thus D is said to *loosely detect* v if an only if for every program P, $D(P)$ terminates, returns *true* if P is indeed infected with v and returns something other than *true* if P is not infected with any virus. The procedure D however may return any result at all for programs infected with some virus different from v (but the program terminates).

As a practical consequence, Chess and White dispel the notion that it is always possible to create a detection procedure for a given virus that has no false positive, even if you have a copy of the virus at your disposal's. The conclusion of their paper identified as an open problem the formal characterization of their concept of looser detection and those of false positives and non-detected cases.

The purpose of the present paper is to solve this problem and to give the formal characterization they asked for. By considering a suitable statistical framework, we first model antiviral detection from a statistical point view and precisely define the inherent aspects and limitations of practical detection. Moreover, whereas Chess and White did only give existence results (and a trivial example of an undetectable virus), we define the concept of statistical simulability and thus characterizes the different practical ways to effectively build such undetectable viruses. With respect to this particular point, we give some practical applications.

The main interest of our study is not to identify suitable techniques to build undetectable viruses but to have a more powerful framework with respect to antivirus evaluation and testing. Indeed, as stated in [5], detection schemes can be very complex and exhaustive analysis of detection patterns is not tractable. Consequently, the statistical approach developed in the present paper enables to consider antivirus software on a sampling-based or probabilistic viewpoint.

This paper is organized as follows. The first section presents the theoretical tools we use and defines our statistical

E. Filiol (✉) · S. Josse
Ecole Supérieure et d'Application des Transmissions,
Laboratoire de virologie et de cryptologie, B.P. 18,
35998 Rennes Armées, France
e-mail: eric.filiol@esat.terre.defense.gouv.fr

S. Josse
Silicomp - AQL, Cesson Sévigné, France
e-mail: sebastien.josse@esat.terre.defense.gouv.fr

framework. Then, we present our statistical model for viral detection. In particular we prove that false positive probability and non-detection probability cannot be cancelled whatever may be the number of detection tools we use. In the next section, we apply our model to antiviral detection techniques used by commercial antivirus. We then show that any such techniques can be modeled by one or more statistical testings. From our model, we present constructive aspects of undecidable virus writing based on the notion of test simulability. Finally, a statistical variant of Cohen's undecidability result is given.

2 Background and notation

We are going to recall here the statistical tools we use throughout this paper.

2.1 The statistical framework

We consider a statistical variable X which describes a random event under scrutiny. We aim at estimating the main characteristics of its probability law $\mathcal{P}$. For such a given estimation of X on a sample we use an *estimator* which is computed on every sample. So we define a function $f(X_1, X_2, \ldots, X_n)$ whose entries are the different values X_i of X on the sample elements. Let us denote $\theta_n^* = f(X_1, X_2, \ldots, X_n)$ this measure whose role is to estimate an unknown feature θ of $\mathcal{P}$.

In this paper, we will consider the parametric case: $\mathcal{P}$ depends on the distribution of X only. The set Θ which is included in $\mathbb{R}^k$, where k is the dimension of θ (e.g. 1 for a mean and 2 for a variance) is the *parametric space*. In an antiviral context, we will limit this set to $\mathbb{N}^k$. The form of $\mathcal{P}$ is assumed to be known.

Statistical testing aims at deciding between several hypotheses from data collected in one or more samples. These samples have been extracted from the population under scrutiny. In other words, we want to determine which population must be considered among many possible ones.

2.2 Statistical testing

The aim is to decide whether a given hypothesis $\mathcal{H}$ must be rejected or kept. The decision process is based on the sample $(X_1, X_2, \ldots, X_n)$ only and on an assumption on the possible probabilistic law of X. In the general case, we sometimes have to decide between r different hypotheses $\mathcal{H}_1, \mathcal{H}_2, \ldots, \mathcal{H}_r$. With respect to each of them, X is supposed to have $\mathcal{P}_i$ as the distribution law. Without loss of generality, we will limit ourselves to the case where $r = 2$.

A statistical testing consists in keeping or rejecting a hypothesis according to which θ is in a set Θ_0. This hypothesis, denoted $\mathcal{H}_0$ is referred as the *null hypothesis*. It is the assumed value for θ. We then consider an *alternative hypothesis* denoted $\mathcal{H}_1$ for which $\theta \in \Theta_1 = \Theta - \Theta_0$. So a test consists in comparing

$$\mathcal{H}_0 : \theta \in \Theta_0 \quad \text{against} \quad \mathcal{H}_1 : \theta \in \Theta_1.$$

Three different types of testing then exist, according to the nature of sets Θ_0 and Θ_1:

- Both $\mathcal{H}_0$ and $\mathcal{H}_1$ are simple and we have $\Theta = \{\theta_0, \theta_1\}$:

  $$\mathcal{H}_0 : \theta = \theta_0 \quad \text{against} \quad \mathcal{H}_1 : \theta = \theta_1.$$

- $\mathcal{H}_0$ is simple and $\mathcal{H}_1$ is multiple ($|\Theta| > 2$) :

  $$\mathcal{H}_0 : \theta = \theta_0 \quad \text{against} \quad \mathcal{H}_1 : \theta \neq \theta_0.$$

- Both $\mathcal{H}_0$ et $\mathcal{H}_1$ are multiple ($|\Theta| > 2$) :

  $$\mathcal{H}_0 : \theta \in \Theta_0 \quad \text{against} \quad \mathcal{H}_1 : \theta \in \Theta_1.$$

We will only consider the first case since the two other ones can be expressed in terms of the first one. Due to lack of space, we will not address the case of goodness of fit tests. From a conceptual point of view, they do not significantly differ from hypothesis testing, in particular with respect to our concern of an antiviral context.

The main basic tool to build a test is the *estimator*, denoted E. Its observed value computed on a given sample is denoted e. According to the hypotheses of the testing, E has a different probabilistic law. So by comparing the value $E = e$ with that of a *decision threshold*, we keep or reject $\mathcal{H}_0$. This threshold in fact partitions the set of possible values for E in two disjoint sets of $\mathbb{R}$, denoted A (*acceptance region*) and $\overline{A} = \mathbb{R} \backslash A$ (*rejection region*).

Since any decision is based on random samples, two types or error can occur with a given probability (*error probabilities*). Table 1 summarizes all the different error cases. In other words, we have:

Table 1 Probabilities of error attached to statistical testings

Decision	$\mathcal{H}_0$ true	$\mathcal{H}_1$ true
Keep $\mathcal{H}_0$	$1 - \alpha$	β
Reject $\mathcal{H}_0$	α	$1 - \beta$

- the type I error α given by:

  $$\alpha = P[E \in \overline{A} \,|\, \mathcal{H}_0 \text{ true}] = P[e > S \,|\, \mathcal{H}_0 \text{ true}],$$

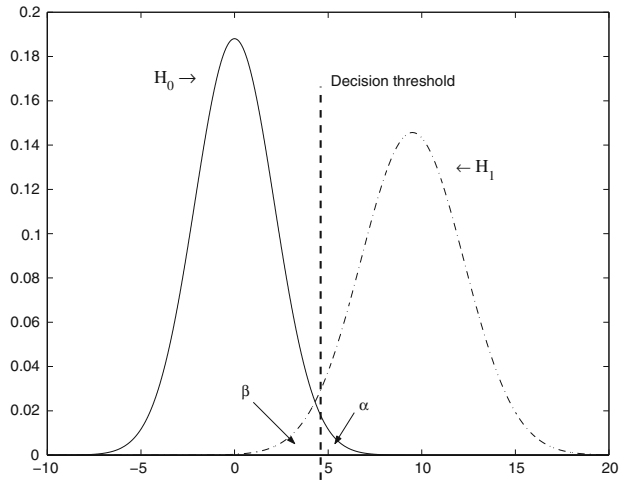

Fig. 1 Statistical model of antiviral detection

and in an antiviral detection context, if the null hypothesis describes the fact that no infection occurred, the type I error represents the false positive probability.

- the type II error β given by:

$$\beta = P[E \in A \mid \mathcal{H}_1 \text{ true}] = P[e < S \mid \mathcal{H}_1 \text{ true}],$$

In an antiviral context, it corresponds to the non-detection probability (if $\mathcal{H}_0$ is the non infection hypothesis).

These two different errors are described in Fig. 1. It is essential to point out the fact that these two error risks are opposite to one from the other. Indeed, each of them depends respectively on A and $\overline{A}$, whose disjoint union is $\mathbb{R}$. If we increase the size of the acceptance region A, automatically that of rejection region $\overline{A}$ decreases.

In most practical testings, the type I error is privileged. Regions A and $\overline{A}$ are directly determined by it. Thus we have to know the law of the null hypothesis. On the contrary, the law of the alternative hypothesis is generally unknown. That implies that determining the exact value of β is most of the time impossible.

To end with statistical testing, it is essential to stress on the fact that error risks α and β are, by definition never equal to zero, except if the $\mathcal{H}_0$ and $\mathcal{H}_1$ probabilistic law domains (density functions) are disjoints. But in the latter case, it becomes obvious that deciding is easy and that no testing is required at all. From a practical point of view, our detailed analysis of commercial antivirus [5] has shown that vendors generally choose to minimize α to the detriment of β.

3 Formal characterisation of antiviral detection

Let us now define our formal characterisation of virus detection.

3.1 Definition of the model

A detection procedure D has to decide whether an input file F is infected or not. For that purpose, D brings statistical testing into play. As shown in [5], most of the time only one is performed but as soon as viral techniques are becoming more and more complex, antivirus should be forced to consider several techniques (e.g. testings) at a time.

Let us assume that detector D performs n statistical testings $\mathcal{T}_1, \ldots, \mathcal{T}_n$. Without loss of generality we will consider that these tests are applied sequentially, each of them applying to the results of the previous one. By definition, every of these testing is marred by type I and II risks α_i and β_i.

Without loss of generality, we assume that the testings T_i are independent one from each other. The result of $\mathcal{T}_i$ does neither affect nor depend on that of $\mathcal{T}_j$ for $i \neq j$. Let us then state our first result.

Proposition 1 *Let n independent statistical testings $\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_n$, with respective type II error β_i. Then the detection procedure D has residual non-detection probability equal to*

$$\beta = \prod_{i=1}^{n} \beta_i.$$

Proof It is sufficient to prove it for $n = 2$. The equivalent cumulated test $\mathcal{T}_{1,2}$ will be considered with testing $\mathcal{T}_3$ and so on.

Let us denote $\mathcal{H}_0^i$ and $\mathcal{H}_1^i$ the null and alternative hypotheses respectively for the test $\mathcal{T}_i$. The probability β is then defined by

$$\beta = P[\text{non-detection}|\mathcal{H}_1^1 \text{ and } \mathcal{H}_1^2 \text{ true}].$$

Under the assumption of independence, the alternative hypotheses of $\mathcal{T}_1$ and $\mathcal{T}_2$ are independent too. We thus have:

$$\beta = P[\text{non-detection}|\mathcal{H}_1^1 \text{ true}] \times P[\text{non-detection}|\mathcal{H}_1^2 \text{ true}].$$

and consequently

$$\beta = \beta_1 \times \beta_2.$$

□

Proposition 1 shows that if the non-detection probability tends to zero, it cannot be equal to zero. Somehow, this is a statitical variant of Cohen's undecidability result. If a large enough family of testings (with $n \to \infty$), would result in detecting with a null residual non-detection probability, this would contradict Cohen's result. As a consequence, as clearly illustrated in Fig. 1, the resulting false positive probability

would be equal to one. Every possible file would be systematically and wrongly detected.

Remark In the general case, testing may be dependent. Then β_1 will affect β_2 and so on. But the previous result holds by considering conditional probabilities. If we denote A_i as the event "non-detection occurs for test i", then

$$P[A_1A_2A_3\ldots A_n] = P[A_1]P[A_2|A_1]P[A_3|A_1A_2]\ldots \times P[A_n|A_1A_2\ldots A_{n-1}].$$

As a general rule, the more testings are independent, the less quickly β tends to zero.

As far as false probability is concerned (risk α), we can state:

Proposition 2 *Let n independent statistical testings $\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_n$, with respective type I error α_i. Then the detection procedure D has residual false positive probability equal to*

$$\alpha = \prod_{i=1}^{n} \alpha_i.$$

Proof Let us prove for $n = 2$. Let us assume that N files are analyzed. Testing $\mathcal{T}_1$ wrongly detects in average $\alpha_1 \times N$ of them. Testing $\mathcal{T}_2$ is then applied on those remaining files. By definition, in average $\alpha_1 \times N \times \alpha_2$ files are wrongly detected. Hence we have a residual $\alpha = \alpha_1 \times \alpha_2$. □

The formal characterisation being defined, let us now precise the two existing case: detecting known codes or know viral techniques and detecting unknown codes using known viral techniques.

3.2 Detection model for known alternative probabilistic law

The knowledge about the alternative law generally comes from the malware analysis. The corresponding model is illustrated by Fig. 1. In order to illustrate this case, let us consider the didactic case of a trivial polymorphic technique: every `XOR Reg, Reg` is replaced with a `MOV Reg, 0` instruction (this is one of the rewriting rules in the metamorphic `Win32/MetaPHOR` engine).

Our sample will be made of N assembly instructions in the code to be analysed. The estimator E measures the frequency of the `MOV Reg, 0` instruction. Let us formulate the null and alternative hypotheses. In a real context, they will be defined through the statistical analysis of a large set of uninfected files ($\mathcal{H}_0$) and infected ones ($\mathcal{H}_1$). We take the theoretical mean value μ of E as θ parameter.

Let us assume that under $\mathcal{H}_0$ the `MOV Reg, 0` appears with probability p_0. Thus, in average[1] $\mu = \mu_0 = N \times p_0$.

On the other hand, experiments have clearly shown that for infected files, the `MOV Reg, 0` instruction occurs with probability p_1 and thus $\mu = \mu_1 = N \times p_1$. By definition, we have $\mu_0 < \mu_1$. We also know the standard deviations (σ_0 and σ_1) with respect to both hypotheses ($\sigma_i = \sqrt{N \times p_0 \times (1 - p_0)}$).

We thus have the following statistical testing:

$$\mathcal{H}_0 : \mu = \mu_0 \quad \text{against} \quad \mathcal{H}_1 : \mu = \mu_1.$$

We will choose $\alpha = 0.05$. Thus, in average 5% of the files being analysed will be wrongly detected. As we assume to known the alternative law, we can fix $\beta = 0,01$. This means that we do not accept that more than 1% of the infected files remain undetected.

Since the central limit theorem applies ($N > 30$), we will use the normal distribution as an approximation of the probabilistic law of $\theta = \mu$ (otherwise we would use the Student law). Thus under $\mathcal{H}_0$, then E follows the $\mathcal{N}(\mu_0, \sigma_0)$ whereas under $\mathcal{H}_1$, it is distributed according to $\mathcal{N}(\mu_1, \sigma_1)$.

All things being now defined, it is somehow easy yet tricky to compute the decision threshold S. Basic calculus techniques (see [7] for the details) give:

$$N_{\min} = \left(\frac{b\sqrt{p_1q_1} - a\sqrt{p_0q_0}}{p_0 - p_1}\right)^2,$$
$$S = Np_0 + a\sqrt{Np_0q_0} = Np_1 + a\sqrt{Np_1q_1}$$

where $q_i = 1 - p_i$ and, with $\Phi(.)$ denoting the normal cumulative density function,

$$a = \Phi^{-1}(\alpha)$$
$$b = \Phi^{-1}(\beta)$$

The value $N_{\min}$ is the minimum value that N (the number of instructions in the code) must have so that our statistical model holds. Otherwise we would either have to consider another estimator or modify the risk values.

The decision rule is then straightforward. Suppose that $E = e$ with respect to the file being analysed, we have:

[1] We will use in a first approach that each instruction is a Bernoulli variable, with the successful event being defined as "is equal to `MOV Reg, 0`". Despite the fact that instructions in the code are not really independent (it is in fact a Markov) process, we will approximate as they would be. Simulations have shown that this approximation is good enough to describe things while keeping the model simple enough.

if $\frac{e-\mu_0}{\sigma_0} < \frac{S-\mu_0}{\sigma_0}$,	we cannot reject $\mathcal{H}_0$,	
	the file is likely to be non infected,	
otherwise,	we reject $\mathcal{H}_0$,	
	the file is likely to be infected.	

3.3 Detection model for unknown alternative probabilistic law

This case refers to the detection of unknown codes that nonetheless use known viral techniques. Thus, we know the null hypothesis only:

$$\mathcal{H}_0 : \theta = \theta_0.$$

We can perform any of the three hypothesis testing:

$$\mathcal{H}_1 : \theta > \theta_0$$
$$\mathcal{H}_1 : \theta < \theta_0$$
$$\mathcal{H}_1 : \theta \neq \theta_0$$

We will consider the last two-sided test which is the most general one (we do not any assumption on the value θ compared to the value θ_0).

Let us consider the estimator E for the appearance frequency of the instruction `XOR Reg, Reg` in uninfected files. We state the null hypothesis as $\mathcal{H}_0 : \mu = \mu_0$. Since many polymorphic/metamorphic techniques use rewriting techniques [7], it is likely that in infected files μ_0 may either decrease (when replaced by the `MOV Reg, 0` instruction) or increase (when using garbage code). So we state the alternative hypothesis as $\mathcal{H}_1 : \mu \neq \mu_0$.

As in the previous section, we assume that $\mathcal{H}_0$ has the normal distribution $\mathcal{N}(\mu_0, \sigma_0)$. The testing is built in the same way as before. We fix $\alpha = 0,05$ (as an example). This corresponds to the rejection region that is divided into two equal parts (recall the test is two-sided; see Fig. 2), each of them referring to a rejection probability equal to $\frac{\alpha}{2}$. But the essential difference with the previous testing lies in the fact that we cannot bring the risk β under control and thus infer the corresponding non-detection probability with respect to the present testing. Consequently, the decision threshold S is such that

$$P\left[\frac{|E-\mu_0|}{\sigma_0} > \frac{S-\mu_0}{\sigma_0}\right] = \alpha.$$

The value $z_{\frac{\alpha}{2}} = \frac{S-\mu_0}{\sigma_0}$ is given by the normal cumulative density function.

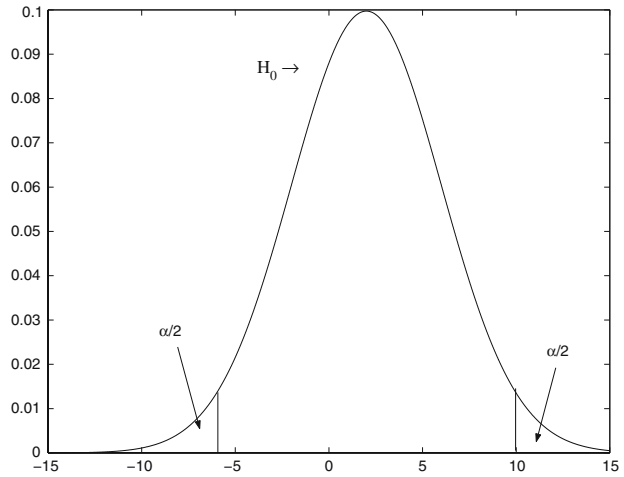

Fig. 2 Statistical model of unknown viruses detection

Table 2 Decision rule for unknown viruses detection testings

if R.C. $< z_{\frac{\alpha}{2}}$,	we cannot reject $\mathcal{H}_0$,	
	the file is likely to be uninfected,	
otherwise,	we reject $\mathcal{H}_0$,	
	the file is likely to be infected.	

Finally, if $E = e$ on the code being analysed, we compute the *critical ratio*:

$$\text{R.C.} = \frac{|e-\mu_0|}{\sigma_0}.$$

The decision rule is then given in Table 2. This kind of testing enables to consider a file as a suspect one ($\mathcal{H}_0$ is rejected). Consequently the file will be systematically disassembled in order to confirm its true nature (infected or not). On the contrary, we cannot have any insight on the non-detection probability. The present model clearly explains why.

4 Antiviral detection techniques and statistical testing

Let us now explain how classical antiviral detection techniques can be modeled as statistical testings.

4.1 Classical signature-based detection

Most of the time, antivirus software heavily relies on pattern detection techniques [5]. In other words, for a malware $\mathcal{M}$, they look for a fixed sequence of bytes $\sigma_{\mathcal{M}} \in \{0, 1, 2, \ldots, 255\}^s$ with $s = |\sigma_{\mathcal{M}}|$.

Let us consider a file F to be analyzed by means of a detector D. We then model this technique by considering that most of the pattern search algorithms (Rabin–Karp algorithm [9], finite automaton search [1], Knuth–Morris–Pratt method [10], Boyer–Moore algorithm [2]...) evaluate a large number

of s-byte subsets in F, before to eventually find the suitable viral pattern.

The search for a viral pattern $\sigma_{\mathcal{M}}$ may be described by a sum of Bernoulli trials. We define the following Bernoulli variable[2] with respect to a subset $\mathcal{S}_i$ of bytes of size s (non necessary contiguous) in F.

$$X_{\mathcal{S}_i} = \begin{cases} \mathcal{S}_i = \sigma_{\mathcal{M}} & p_0 \\ \mathcal{S}_i \neq \sigma_{\mathcal{M}} & q_0 = 1 - p_0 \end{cases}$$

The value p_0 is directly determined by its efficiency as defined in [5]. If the file F contains n bytes, there are at most $N = \binom{n}{s}$ possible s-byte subsets in F. Then we define the following estimator:

$$Z = \sum_{\mathcal{S}_i \in \mathcal{P}(F), |\mathcal{S}_i| = s} X_{\mathcal{S}_i}.$$

Then the null hypothesis $\mathcal{H}_0$ states that F is not infected. The relevant probabilistic law is the normal distribution $\mathcal{N}(N.p_0, \sqrt{Np_0q_0})$. The alternative hypothesis $\mathcal{H}_1$ is very simple to determine. We indeed have $\mathcal{H}_1 : \mu = 1$. If the file F is infected, the detector will terminate as soon as it has found the viral pattern $\sigma_{\mathcal{M}}$. Consequently, we can describe the "law" of $\mathcal{H}_1$ by the density function $f(1) = 1$ and $f(x) = 0$ whenever $x \neq 1$. Then $\mu_1 = 1$ and $\sigma_1 = 0$.

Let us recall that only two cases may occur:

- the file F is infected by the malware $\mathcal{M}$, so we have $\mu = \mu_1 = 1$ (with a probability which is equal to 1);
- the file F is infected by a malware $\mathcal{M}'$ which is not recorded in the viral database yet. Then, with respect to $\sigma_{\mathcal{M}}$, we have $\mu = \mu_0$.

The corresponding statistical testing is then simple to set up. For a fixed α, we compute $z_{\frac{\alpha}{2}}$ and R.C as shown in the previous section (case of unknown alternative probabilistic laws). Finally the suitable decision rule is applied. From a practical point of view, as soon as n and s are large enough, an empiric threshold equal to 1 can be used.

Here computing the non-detection probability β is non sensical. If the file F is infected by the malware $\mathcal{M}$ for which the viral database has been updated, the probability $P[Z < 1|\mathcal{H}_1 \text{ true}] = 0$ with respect to $\mathcal{M}$. Indeed, we have $P[Z \geq 1|\mathcal{H}_1 \text{ true}] = 1$. If the code is a known one, it will be systematically detected. Consequently, to evaluate the non-detection probability as far as signature-based detection is concerned, we must not do it with respect to a given malware. In this respect, we can define the non-detection probability as follows:

$$P[\text{non-detection}] = \frac{|\mathcal{B}_{\mathcal{D}}|}{\aleph_0} = \frac{|\mathcal{B}_{\mathcal{D}}|}{\infty} \to 0,$$

where $\mathcal{B}_{\mathcal{D}}$ describes the viral database used by detector D and where $\aleph_0$ denotes the total number of possible viruses [4, Chap. 3].

4.2 The general case of detection scheme or detection strategies

Detection schemes [5] with respect to a given malware $\mathcal{M}$ is a pair $\{\mathcal{S}_{\mathcal{M}}, f_{\mathcal{M}}\}$ where $\mathcal{S}_{\mathcal{M}}$ is the detection pattern and $f_{\mathcal{M}}$ is the detection function. The concept of detection strategies [6] generalizes that of detection scheme by considering a triplet $\mathcal{DS} = \{\mathcal{S}_{\mathcal{M}}, \mathcal{B}_{\mathcal{M}}, f_{\mathcal{M}}\}$ which additionally includes a set $\mathcal{B}_{\mathcal{M}}$ of program functions (behaviors). It has been proved that whereas detection scheme models every possible sequence-based detection techniques [5], detection strategies models general detection [6], whatever techniques is considered – sequence-based and/or behavior-based.

In this general context, the previous statistical model fully applies. The essential difference lies in the fact that the detection function has a weight which is strictly greater than 1—simple signature based detection considers detection function which has a weight strictly equal to 1.

Let us note $\omega = \text{wt}(f)$, the expected probability p_0 equals $\frac{\omega}{256^s}$ instead of $\frac{1}{256^s}$. Consequently the false positive probability increases, but most of the time it remains negligible.

4.3 Other cases

For a number of (exotic) detection techniques, the detection function is too complex to be considered. As a consequence, stating the null hypothesis is practically untractable and the exact distribution law cannot be determined. An empiric law is then to be considered instead, which can be tuned up according a posteriori knowledge (Bayesian approach).

The authors of new techniques generally give experimental results as far as risks α and β are concerned. As an example, self-organizing map-based detection technique, which has been proposed by I. Yoo and U. Ultes-Nitsche [13], yields $\alpha = 0,3$ and $\beta = 0,16$. Every new technique proposed should be systematically published with those experimental risk values along with all relevant data that would help to reproduce that technique.

Due to lack of space, we will not go deal with heuristics [11,12] or meta-heuristics like tabu search, greedy algorithms, simulated annealing... From a general point of view, we have shown in [7] that any heuristic or meta-heuristics can be translated in terms of statistical tests. Consequently, all previous results hold for these particular detection techniques.

[2] In a first approach, this approximation holds since no particular assumption is done about files F to be analysed. Thus they may be considered a priori as containing random data.

5 Statistical testing simulability: defeating antiviral detection

Our formal characterization of antiviral detection clearly enables us to precisely identified why and how these techniques have inherent limitations. But beyond the intrinsic risk of wrong decisions, there exist a much worse situation: when the attacker use the detection techniques against the defender. As soon as the first one knows which tools are used by the second, he is able to perform what we call *statistical testing simulability*.

Let us give a definition for this concept.

Definition 1 Simulating a statistical testing consists for an adversary, to introduce, in a given population $\mathcal{P}$, a statistical bias that cannot be detected by an analyst by means of this test.

There exist two different kinds of simulability:

- the first one does not depend on the testings (and their parameters) the defender usually considers. It is called *strong testing simulability*.
- on the contrary, the second one does depend on those testings that the attackers aims at simulating. It is called *weak testing simulability*.

In this section, we call "tester" the one who uses statistical testing in order to decide.

5.1 Strong testing simulability

Let us define this concept more precisely.

Definition 2 (Strong testing simulability) Let P be a property and T a testing whose role is to decide whether P holds for given population $\mathcal{P}$. Strongly simulating this testing consists in modifying or building a biased population P in such a way that T systematically decides that P holds on $\mathcal{P}$, up to the statistical risks. But there exists a statistical testing T' which is able to detect that bias in $\mathcal{P}$. In the same way, we say that t testings $(T_1, T_2, \ldots, T_t)$ are strongly simulated, if applying them results in deciding that P holds on $\mathcal{P}$ but does no longer hold when considering a $(t+1)$-th testing T_{t+1}.

In terms of security, strong simulability is a critical aspect in security analysis. Many applications have been identified, especially in cryptography [7]. In an antiviral context, strong simulability exist when the malware writer, who has identified any of the techniques used by one or more antivirus, writes malware that cannot be detected by the target antivirus but only by the malware writer. As a typical example, a viral database which contains t signatures is equivalent to t testings (see previous section) and any new malware corresponds to the testing T_{t+1}.

5.2 Weak testing simulability

Let us define this concept more precisely.

Definition 3 (Weak testing simulability) Let P be a property and T a testing whose role is to decide whether P is valid for a given population $\mathcal{P}$ or not. To weakly simulate this testing means introducing in $\mathcal{P}$ a new property P' which partially modifies the property P, in such a way that T systematically decides that P holds on $\mathcal{P}$, up to the error risks.

Weak simulability differs from strong simulability since the attacker considers the same testings as the tester. The attacker thus introduces a bias that the tester used will not be able to detect.

The property P' of Definition 3 is generally opposite to the property P. It precisely represents a flaw that the attacker aims at exploiting. Bringing weak simulability into play is somehow tricky. It requires to get a deep knowledge of the testings to be simulated.

The central approach consists in introducing the property P' in such a way that the estimators E_i in use remain in the acceptance region of the testing (generally that of the null hypothesis). Let us recall that during the decision step, the tester checks whether $E < S$ or not. Thus weak simulability consists in changing the value $S - E$ while keeping it positive. For that purpose, we use the intrinsic properties of the relevant sampling distribution.

5.3 Application: defeating detection

Many applications of testing simulability have been imagined [7]:

- bypassing stream filtering (e.g. to detect the spread of worms);
- bypassing data filtering (e.g. blocking of encrypted or zipped data);
- defeating antiviral detection.

Let us present a detailed case with respect to the last point. It is essential to point out that simulability is done by the attacker once and for all. Indeed, antivirus publishers do not change the underlying statistical model of their products very often. Without loss of generality, we will consider the case of spectral analysis which seems the more tricky to simulate since the underlying detection function is too complex to represent.

Let us recall that spectral analysis [4] consists in listing some critical instructions of the code. This list is called the *spectrum*. Every compiler used a small subset only of all possible intructions. On the contrary, a malware will consider the whole set of these instructions.

Let us consider a spectrum with respect to a given compiler $\mathcal{C}$ made of a list of instructions $(I_i)_{1\leq i\leq c}$, each of them along with its expected frequency n_i (in uninfected programs). The spectrum is thus given by:

$$spec_j(\mathcal{C}) = (I_i, n_i)_{1\leq i\leq c}.$$

The index j means that in practice we may consider more than one spectrum. Without loss of generality, we will consider only one.

Whenever a program is analysed by the detector, the observed frequencies $\hat{n}_i$ are recorded. The null hypothesis $\mathcal{H}_0$ is the statement:

$$\mathcal{H}_0 : \hat{n}_i = n_i \quad 1 \leq i \leq c.$$

The alternative hypothesis $\mathcal{H}_1$ consists in claiming that the spectrum of the program being analysed significantly differs from the expected one and thus the program is likely to be infected.

We use then the estimator

$$D^2 = \sum_{i=1}^{c} \frac{(\hat{n}_i - n_i)^2}{n_i}.$$

It is a well-known fact that D^2 asymptotically has a χ^2 distribution with $c-1$ degrees of freedom, under the assumption that $\mathcal{H}_0$ holds. For a fixed type I error α, the decision step consists in comparing the value of estimator D^2 with the decision threshold $\chi^2_{(\alpha,c-1)}$. We then decide:

$$\begin{cases} \mathcal{H}_0 & \text{if } D^2 \leq \chi^2_{(\alpha,c-1)} \\ \mathcal{H}_1 & \text{if } D^2 > \chi^2_{(\alpha,c-1)}. \end{cases}$$

Let us now explain how to simulate this testing. Let us assume that during a metamorphic process in a malware, rewriting techniques have been used. They are likely to modify the expected frequency n_i in the reference spectrum. The metamorphic engine generally considers random rewriting rules [7]. But how random must be the rules in order to not trigger any alert when the file is analysed?

Without loss of generality and for the sake of clarity, let us consider that only two instructions I_{i_1} and I_{i_2} of the spectrum have been rewritten. Let us explain how the metamorphic engine will have to tune up its mutations in order to weaky simulate spectral analysis.

Instructions I_{i_1} and I_{i_2} have expected frequency n_{i_1} and n_{i_2} respectively. Let us first consider the initial code (variant at time $t = 0$). It is such that

$$D^2 = \sum_{i=1}^{c} \frac{(\hat{n}_i - n_i)^2}{n_i} \leq \chi^2_{(\alpha,c-1)}, \tag{1}$$

with respect to a given type I error α. The corresponding initial (reference) value of D^2 is denoted D_0^2 and we denote $F = \chi^2_{(\alpha,c-1)} - D^2$.

Then the metamorphic process modifies frequency $\hat{n}_{i_1}$ and $\hat{n}_{i_2}$ as follows:

$$\hat{n}_{i_1} \to \hat{n}'_{i_1} \qquad \hat{n}_{i_1} + \delta_1 = \hat{n}'_{i_1} \tag{2}$$

$$\hat{n}_{i_2} \to \hat{n}'_{i_2} \qquad \hat{n}_{i_2} + \delta_2 = \hat{n}'_{i_2} \tag{3}$$

Values δ_i may be negative or positive. In a first approach, it is obvious that a first criterion is to make sure that $E[\delta_1 + \delta_2] = 0$ (the mathematical mean of instruction modifications is equal to 0). But this criterion is rather restrictive and the number of mutation possibilities will be limited. Moreover, this criterion depends on the initial code only and not on the expected frequencies n_i. Let us consider a more general and more powerful criterion.

Copying out the modifications of $\hat{n}_{i_1}$ and $\hat{n}_{i_2}$ in Equation (1), we can write

$$D^2 = \sum_{i=1, i\notin\{i_1,i_2\}}^{c} \frac{(\hat{n}_i - n_i)^2}{n_i} + \frac{(\hat{n}_{i_1} - n_{i_1})^2}{n_{i_1}} + \frac{(\hat{n}_{i_2} - n_{i_2})^2}{n_{i_2}}. \tag{4}$$

This enables to come back to the value D_0^2, when taking Equations (2) and (3) into account:

$$D^2 = D_0^2 + \frac{2\delta_1(\hat{n}_{i_1} - n_{i_1}) + \delta_1^2}{n_{i_1}} + \frac{2\delta_2(\hat{n}_{i_2} - n_{i_2}) + \delta_2^2}{n_{i_2}}. \tag{5}$$

Equation (5) thus gives the criterion we are looking for:

$$E\left[\frac{2\delta_1(\hat{n}_{i_1} - n_{i_1}) + \delta_1^2}{n_{i_1}} + \frac{2\delta_2(\hat{n}_{i_2} - n_{i_2}) + \delta_2^2}{n_{i_2}}\right] = 0. \tag{6}$$

or equivalently

$$E[2\delta_1(\hat{n}_{i_1} - n_{i_1})n_{i_2} + \delta_1^2 n_{i_2} + 2\delta_2(\hat{n}_{i_2} - n_{i_2})n_{i_1} + \delta_2^2 n_{i_1}] = 0. \tag{7}$$

A quick analysis shows that this criterion is far less restrictive that the first one ($E[\delta_1 + \delta_2] = 0$).

The polymorphic engine has thus to randomly modify instructions I_{i_1} and I_{i_2} in such a way that frequencies $\hat{n}_{i_1}$ and $\hat{n}_{i_2}$ keep on verifying the criterion stated by Equation (7).

This simulability scheme can be generalised further. We only considered here a didactic, simple scenario. Let us give some more sophisticated mechanisms:

- the criterion of Equation (7), from a practical point of view, must be brought into play along with the variance given by

$$\epsilon = 2\delta_1(\hat{n}_{i_1} - n_{i_1})n_{i_2} + \delta_1^2 n_{i_2} + 2\delta_2(\hat{n}_{i_2} - n_{i_2})n_{i_1} + \delta_2^2 n_{i_1}.$$

Indeed, weak simulability practically requires that

$$D_0^2 + \epsilon < \chi^2_{(\alpha,c-1)},$$

or equivalently that $\epsilon < F$. If we would take the value $E[\epsilon]$ only into account, with a non null probability, the value $\chi^2_{(\alpha,c-1)}$ may be too large thus resulting in rejecting $\mathcal{H}_0$. We thus are forced to consider the variance (in other words the mean behaviour of the deviations with respect to $E[\epsilon]$);

- it is powerful to consider a large enough spectrum, Indeed it allows to create fake instructions modifications that can balance other useful or compulsory modifications. It is thus easier for the estimator to remain in the acceptance region;
- the value ϵ, when generalised to a larger spectrum, can also have a negative statistical mean ($-D_0^2 \leq \epsilon \leq F$). This increases the power of weak simulability .

6 A statistical variant of Cohen's undecidability

We have proved with Proposition 1 that non-detection probability cannot be equal to 0 in any way. This result represents somehow a statistical formalisation Cohen's undecidability [3]. Indeed, if a finite enumerably set of statistical tests would cancel the non-detection probability, thus we would obtain a result that contradicts Cohen's one. As far as false positive probability is concerned, according to Fig. 1, it would equal exactly one. In other words, we would detect any file submitted to detection as infected.

The concept of test simulability enables us to give an illustration of the undecidable antiviral detection which is a statistical equivalent to Cohen's famous algorithmic example called *contradictory virus* (CV). Let us consider a detection procedure D which is able to distinguish a virus V from any other program. Its pseudo-code is given by:

```
CV() {
   ......
   main() {
     if not D(CV) then {
       spread();
       if trigger_condition is true then payload();
      }
     End if
     Go to next target program
    }}
```

F. Cohen has proved that D did not exist and that any detection based on D is undecidable.

According to our statitical model of detection and to the concept of statistical simulability we have developed, the virus CV may be reformulated as follows. The detection procedure D is made of n statistical tests $\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_n$. Then, CV's pseudo-code is given by:

```
CV() {
   ......
  simul(D, V) {
     generate V' by (statistically) simulating D from V
     return V'
    }

   main() {
     if not D(CV) then {
       spread();
       if trigger_condition is true then payload();
      }
     else
      {
       V' = simul(D, CV)
       run V'
      }
     End if
     Go to next target program
    } }
```

Thus, D is a contradictory procedure as well:

- If D decides that CV is a virus, then CV mutates by simulating tests $\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_n$ in D.
- On the other hand, if D decides conversely (CV is not a virus), the infection occurs and CV is indeed a virus.

7 Future work and conclusion

In this paper we have presented the formal characterisation of antiviral detection. We thus are able to understand why antivirus software have a limited efficiency by nature but how an attacker can exploit antivirus intrinsic limitations. In particular, this model clearly shows that antivirus efficiency directly depends on the antivirus publishers' strategic choices: either lowering false positive probability (but increasing non-detection probability) or lowering non-detection probability (but increasing false positive probability).

Future work addresses the problem of finding more powerful statistical testings scheme that:

- succeeds in lowering as much as possible the type II error with a number of testings which is as much reduced as possible. An interesting trend could be to characterise uninfected programs (thus the null hypothesis) on a more combinatorial basis (e.g use of randomized design, design of experiments);
- and hinders as much as possible testing simulability.

Testing simulability clearly shows that whenever an attacker knows all the detection techniques which are used by an antivirus software, he is able to bypass and to defeat it. This fact strongly militates against detection scheme extraction capabilities of the attacker. Then secure detection scheme as proposed in [5] should be systematically considered and implemented in antivirus software to hinder or even forbid test simulability and thus the ability by attacker to bypass antiviral detection.

References

1. Aho, A.V., Hopcroft, J.E., et Ulman, J.D.: The Design and Analysis of Computer Algorithms. Addison-Wesley, Reading (1974)
2. Boyer, R.S., et Moore, J.S.: A fast string-searching algorithm. Commun ACM **10**(20), 762–772 (1977)
3. Cohen, F.: Computer viruses. Ph.D. Thesis, University of Southern California, Janvier (1986)
4. Filiol, E.: Computer Viruses: from Theory to Applications, IRIS International Series, Springer, France. ISBN 2-287-23939-1 (2005)
5. Filiol, E.: Malware pattern scanning schemes secure against black-box analysis. In: Broucek, V., Turner, P. (eds.) EICAR Conference Best Paper Proceedings, Hamburg, Germany: EICAR. An extended version has been published in the EICAR 2006 Special Issue, J. Comput. Virol. **2**(1), pp. 35–50 (2006)
6. Filiol, E., Jacob, G., Le Liard, M.: Evaluation methodology and theoretical model for antiviral behavioural detection strategies. In: Bonfante, G., Marion, J.-Y. (eds) WTCV'06 Special Issue, J. Comput. Virol. **3**(1) (2007)
7. Filiol, E.: Techniques virales avancées, IRIS Series, Springer, France. An English translation is pending (due mid 2007) (2007)
8. Chess, D.M., White, S.R.: An undetectable computer virus. In: Virus Bulletin Conference (2000)
9. Karp, R.M., et Rabin, M.O.: Efficient Randomized Pattern-Matching Algorithms, Technical report TR-31-81. Ayken Computation Laboratory, Harvard University (1981)
10. Knuth, D.E., Morris, J.H., et Pratt, V.R., Jr.: Fast pattern-matching in strings. SIAM J. Comput. **2**(6):323–350 (1977)
11. Schmall, M.: Heuristische Viruserkennung. Diplom Thesis, Universität Hamburg (1998)
12. Schmall, M.: Classification and identification of malicious code based on heuristic techniques utilizing Meta languages. Ph.D. Thesis, University of Hamburg (2003)
13. Yoo, I., Ultes-Nitsche, U.: Non-signature-based virus detection: towards establishing unknown virus detection technique using SOM. J. Comput. Virol. **3**(2), 163–186 (2006)