SPRT for SPIT: Using the Sequential Probability Ratio Test for Spam in VoIP Prevention

Tobias Jung[1], Sylvain Martin[1], Damien Ernst[1], and Guy Leduc[1]
{tjung,sylvain.martin,dernst,guy.leduc}@ulg.ac.be

Montefiore Institute, University of Liège, Belgium

Abstract. This paper presents the first formal framework for identifying and filtering SPIT calls (SPam in Internet Telephony) in an outbound scenario with provable optimal performance. In so doing, our work deviates from related earlier work where this problem is only addressed by ad-hoc solutions. Our goal is to rigorously formalize the problem in terms of mathematical decision theory, find the optimal solution to the problem, and derive concrete bounds for its expected loss (number of mistakes the SPIT filter will make in the worst case). This goal is achieved by considering a scenario amenable to theoretical analysis, namely SPIT detection in an *outbound* scenario with *pure* sources. Our methodology is to first define the cost of making an error, apply Wald's sequential probability ratio test, and then determine analytically error probabilities such that the resulting expected loss is minimized. The benefits of our approach are: (1) the method is optimal (in a sense defined in the paper); (2) the method does not rely on manual tuning and tweaking of parameters but is completely self-contained and mathematically justified; (3) the method is computationally simple and scalable. These are desirable features that would make our method a component of choice in larger, autonomic frameworks.

1 Introduction

SPIT is an acronym for spam in internet telephony and refers to unsolicited calls that, when answered by a human, would deliever a pre-recorded message (e.g., advertisement or phishing attempts). Similar to spam in emails, SPIT exploits the openness of the existing infrastructure (e.g., no strongly authenticated identities) together with the fact that VoIP calls can be easily generated automatically and at zero costs. Unlike with spam in emails however, where the content consists of text and can be analyzed before it is delivered, the content of a phone call (a voice stream) is only available when the call is answered. Thus many of the defensive measures that are effective against email spam do not directly translate to SPIT mitigation. Previously, some first ideas have already been suggested to address this problem. They range from reputation-based [3, 1] and call-frequency based [10] dynamic black-listing, fingerprinting [15], to challenging suspicious calls by captchas [9, 11, 8], or the use of standard machine learning such as anomaly detection [6, 4], clustering [14], or decision trees [5].

These methods provide interesting building blocks, but, in our opinion, suffer from two main shortcomings. First, they do not provide performance guarantees in the sense that it is difficult to get an estimate of the number of SPIT calls that will go through and the number of regular calls they will erroneously stop. Second, they require a lot of hand-tuning for working well, which cannot be sustained in future's networks.

Our initial motivation for this paper was to explore whether there would be ways to design SPIT filters that would not suffer from these two shortcomings. For this, we start by considering an abstracted scenario amenable to theoretical analysis where we make essentially two simplifying assumptions: (1) we are dealing with *pure source* SPIT detection in an *outbound* scenario; (2) we can extract features from calls (such as, for example, call duration) whose distribution for SPIT and regular calls is *known in advance*. Here, "outbound scenario" means that our SPIT detector will be located in, or at the egde of, the network where the source resides, and will check all outgoing calls originating from within the network. Technically, this means that we are able to easily map calls to sources and that we can observe multiple calls from each source. By "pure source" we mean that a source either produces only SPIT or only NON-SPIT calls for a certain observation horizon. Under these assumptions, we have been able to design a SPIT filter which requires no tuning and no user feedback and which is optimal in a sense that will be defined later in this paper. Note that in the paper we will introduce this SPIT filter from a theoretical point of view and, due to the lack of the space, will have to omit most of the issues pertaining to a real-world deployment.

The paper is structured as follows: we start in Section 2.1 with describing precisely and in mathematical terms the context for which we will design the SPIT filter. In Section 2.2 we then show how it is possible to design from a simple statistical test a SPIT filter with the desired optimality guarantees and for which in Section 2.3 we can provide analytical expressions to bound its worst-case performance. In Section 2.4 we then give a concrete example and study the case where the distribution of the relevant call features is an exponential distribution (which would be the case if we chose call duration as feature). Finally, in Section 3 we examine the performance of the SPIT filter with some real-world data and compare the empirical performance with what the theory predicts.

2 A SPIT filter with theoretically optimal performance

2.1 Problem statement

The SPIT filter is placed in the outbound network and monitors outgoing calls originating from a source within the protected network. (The sources correspond to known users.) We assume that sources are independent and over a certain observation horizon will either *only produce regular calls* (if it is a human) or *only produce SPIT calls* (if the source is compromised by a SPIT bot). The purpose of the SPIT filter then is to decide, for each source individually, if it is a regular user or a SPIT bot.

Calls are processed sequentially. Every time a new call arrives at the filter, it can do one of two things: (i) accept the call and pass it on to the recipient or (ii) block the call. Each call is associated with certain features; we assume that the features which are relevant are only observable *after a call is accepted* (e.g., call duration). Furthermore, the distribution over the features shall be different depending on whether the call is SPIT or not. Finally we assume that these distributions are *known in advance*. Our goal is to decide for a source, after observing a few calls from it, whether or not the source sends out SPIT. More precisely, we look for a decision policy that initially accepts all calls, thus refining the belief about whether or not the source is SPIT, and then at some point decides to either block or accept all future calls from the source. Seen as a single-state decision problem over time, the SPIT filter has three possible actions: (1) accept the next call, which reveals its features and thus refines the belief about the type of the source, (2) block all future calls, and (3) accept all future calls. The last two actions immediately stop the decision-making process and, if the decision was wrong (that is, deciding to accept when in truth the source is SPIT, or deciding to block when in truth the source is NON-SPIT), will incur a terminal loss proportional to the number of remaining calls. In addition, we also have a per step cost during the initial exploration if the source is of type SPIT (for erroneously accepting SPIT calls). In doing so, we arrive at a well defined concept of loss. Within the framework outlined above, every conceivable SPIT filter algorithm will have a performance number: its expected loss. The particular SPIT filter that we are going to describe below will be one that minimizes this expected loss.

To address the problem mathematically, we employ Wald's sequential probability ratio test for simple hypotheses introduced in [12]. The sequential probability ratio test (SPRT) has the remarkable property that among all sequential tests procedures it minimizes the expected number of samples for a given level of certainty and regardless of which hypothesis is true (the optimality of SPRT was proved in [13]). In addition, the SPRT comes with bounds for the expected stopping time and thus allows us to derive concrete expressions for the expected loss as a function of the characteristics of the particular problem (meaning we can express the loss as a function of the parameters of the distribution for SPIT or NON-SPIT). Finally, SPRT requires only simple algebraic operations to carry out and thus is easy to implement and computationally cheap to run.

2.2 SPIT detection via the SPRT

To model the SPIT detection problem with the SPRT, we now proceed as follows: Assume we can make sequential observations from one source of *a priori* unknown type SPIT or NON-SPIT. Let x_t denote the features of the t-th call we observe, modeled by random variable X_t. The X_t are i.i.d. with common distribution (or density) p_X. The calls all originate from one source which can either be of type SPIT with distribution $p_{\text{SPIT}}(x) = p(x|\text{source=SPIT})$ or of type NON-SPIT with distribution $p_{\text{NON-SPIT}}(x) = p(x|\text{source=NON-SPIT})$. Initially, the type of the source we are receiving calls from is not known; in absence of other

information we have to assume that both types are equally likely, thus the prior would be $p(\text{SPIT}) = p(\text{NON-SPIT}) = \frac{1}{2}$. In order to learn the type of the source, we observe calls $x_1, x_2, \ldots$ and test the hypothesis

$$H_0 : p_X = p_{\text{SPIT}} \qquad \text{versus} \qquad H_1 : p_X = p_{\text{NON-SPIT}}. \tag{1}$$

At time t we observe x_t. Let

$$\lambda_t := \frac{p(x_1, \ldots, x_t|\text{NON-SPIT})}{p(x_1, \ldots, x_t|\text{SPIT})} = \prod_{i=1}^{t} \frac{p(x_i|\text{NON-SPIT})}{p(x_i|\text{SPIT})} \tag{2}$$

be the ratio of the likelihoods of each hypothesis after t observations $x_1, \ldots, x_t$. Since the X_i are independent we can factor the joint distribution on the left side to obtain the right side. In practice it will be more convenient for numerical reasons to work with the log-likelihoods. Doing this allows us to write a particular simple recursive update for the log-likelihood ratio $\Lambda_t := \log \lambda_t$, that is

$$\Lambda_t := \Lambda_{t-1} + \log \frac{p(x_t|\text{NON-SPIT})}{p(x_t|\text{SPIT})}. \tag{3}$$

After each update we examine which of the following three cases applies and act accordingly:

$$A < \lambda_t < B \implies \text{continue monitoring} \tag{4}$$
$$\lambda_t \geq B \implies \text{accept } H_1 \text{ (decide NON-SPIT)} \tag{5}$$
$$\lambda_t \leq A \implies \text{accept } H_0 \text{ (decide SPIT)} \tag{6}$$

Thresholds A and B with $0 < A < 1 < B < \infty$ depend on the desired accuracy or error probabilities of the test:

$$\alpha := P\{\text{accept } H_1 \,|\, H_0 \text{ true}\} = P\{\text{decide NON-SPIT} \,|\, \text{source=SPIT}\} \tag{7}$$
$$\beta := P\{\text{reject } H_1 \,|\, H_1 \text{ true}\} = P\{\text{decide SPIT} \,|\, \text{source=NON-SPIT}\}. \tag{8}$$

Note that α and β need to be specified in advance such that certain accuracy requirements are met (see next section where we consider the expected loss of the procedure). The threshold values A and B and error probabilities α and β are connected in the following way

$$\beta \leq A(1-\alpha) \qquad \text{and} \qquad \alpha \leq (1-\beta)/B. \tag{9}$$

Note that the inequalities arise because of the discrete nature of making observations (i.e., at $t = 1, 2, \ldots$) which results in λ_t not being able to hit the boundaries exactly. In practice we will neglect this excess and treat the inequalities as equalities:

$$A = \beta/(1-\alpha) \qquad \text{and} \qquad B = (1-\beta)/\alpha. \tag{10}$$

Let T be the random time at which the sequence of the λ_t leaves the open interval (A, B) and a decision is made that terminates the sampling process.

(Note that stopping time T is a random quantity due to the randomness of the sample generation.) The SPRT provides the following pair of equalities for the expected stopping time in both cases (cf. [12], Eqs.(4.80),(4.81))

$$E_{X_i \sim p_{\text{SPIT}}}[T] = \frac{1}{\kappa_0}\left(\alpha \log \frac{1-\beta}{\alpha} + (1-\alpha)\log \frac{\beta}{1-\alpha}\right) \tag{11}$$

$$E_{X_i \sim p_{\text{NON-SPIT}}}[T] = \frac{1}{\kappa_1}\left(\beta \log \frac{\beta}{1-\alpha} + (1-\beta)\log \frac{1-\beta}{\alpha}\right). \tag{12}$$

The constants κ_i with $\kappa_0 < 0 < \kappa_1$ are the Kullback-Leibler information numbers defined by

$$\kappa_0 = E_{x \sim p_{\text{SPIT}}}\left[\log \frac{p(x|\text{NON-SPIT})}{p(x|\text{SPIT})}\right] \tag{13}$$

$$\kappa_1 = E_{x \sim p_{\text{NON-SPIT}}}\left[\log \frac{p(x|\text{NON-SPIT})}{p(x|\text{SPIT})}\right]. \tag{14}$$

The constants κ_i can be interpreted as a measure of how difficult it is to distinguish between p_{SPIT} and $p_{\text{NON-SPIT}}$. The smaller they are the more difficult is the problem.

2.3 Theoretical performance of the SPIT filter

We will now look at the performance of our SPIT filter and derive expressions for its expected loss. Let us assume we are going to receive a finite number N of calls and that N is sufficiently large such that the test will always stop before the calls are exhausted.

How does the filter work? At the beginning all calls are accepted until the test becomes sufficiently certain about its prediction. Once the test becomes sufficiently certain, based on the outcome the filter implements the following simple policy: if the test returns that the source is SPIT then all future calls from it will be blocked. If the test says that the source is NON-SPIT then all future calls from it will be accepted. Since the decision could be wrong, we define the following costs: $c_0 > 0$ is the cost for erroneously accepting a SPIT call, and $c_1 > 0$ is the cost for erroneously blocking a NON-SPIT call. Let L denote the loss incurred by this policy (note that L is a random quantity). To compute the expected loss, we have to divide N into two parts: the first part from 1 to T corresponds to the running time of the test where all calls are automatically accepted ($T < N$ being the random stopping time with expectation given in Eqs. (11)-(12)), the second part from $T+1$ to N corresponds to the time after the test.

If H_0 is true, i.e., the source is SPIT, the loss L will be the random quantity

$$L|\text{source=SPIT} \;=\; c_0 T + \alpha c_0 (N - T) \tag{15}$$

where $c_0 T$ is the cost of the test, α the probability of making the wrong decision, and $c_0(N-T)$ the cost of making the wrong decision for the remaining calls.

Taking expectations gives

$$E_{X_i \sim p_{\text{SPIT}}}[L] = \alpha c_0 N + c_0(1-\alpha) E_{X_i \sim p_{\text{SPIT}}}[T]. \tag{16}$$

Likewise, if H_1 is true, i.e., the source is NON-SPIT, our loss will be the random quantity

$$L|\text{source=NON-SPIT} \;=\; 0 \cdot T + \beta c_1 (N - T) \tag{17}$$

where 0 is the cost of the test (because accepting NON-SPIT is the right thing to do), β the probability of making the wrong decision, and $c_1(N-T)$ the cost of making the wrong decision for the remaining calls. Taking expectation gives

$$E_{X_i \sim p_{\text{NON-SPIT}}}[L] = \beta c_1 (N - E_{X_i \sim p_{\text{NON-SPIT}}}[T]). \tag{18}$$

The total expected loss takes into consideration both cases and is simply

$$E[L] = p(\text{SPIT}) \cdot E_{X_i \sim p_{\text{SPIT}}}[L] + p(\text{NON-SPIT}) \cdot E_{X_i \sim p_{\text{NON-SPIT}}}[L]. \tag{19}$$

For the case that both priors are equal, we have

$$E[L] = \frac{1}{2}\Big\{ N(\alpha c_0 + \beta c_1) + \log\frac{1-\beta}{\alpha}\Big(\frac{c_0\alpha(1-\alpha)}{\kappa_0} - \frac{c_1\beta(1-\beta)}{\kappa_1}\Big) + \log\frac{\beta}{1-\alpha}\Big(\frac{c_0(1-\alpha)^2}{\kappa_0} - \frac{c_1\beta^2}{\kappa_1}\Big)\Big\}. \tag{20}$$

Looking at Eq. (20) we see that, given all the other information, the expected loss will be a function of α, β. In practice, one way of choosing α, β would be to look for that setting α^*, β^* that will minimize the expected loss under the given problem specifications (i.e., distributions $p_{\text{SPIT}}, p_{\text{NON-SPIT}}$ and cost c_0, c_1).

2.4 Example: Exponential duration distribution

For the following numerical example we assume that p_{SPIT} and $p_{\text{NON-SPIT}}$ are both exponential distributions with parameters $\lambda_0, \lambda_1 > 0$, that is, are given by

$$p(x|\text{SPIT}) = \lambda_0 \exp(-\lambda_0 x), \qquad p(x|\text{NONSPIT}) = \lambda_1 \exp(-\lambda_1 x) \tag{21}$$

for $x > 0$. While this example is primarily meant to illustrate the behavior of a SPRT-based SPIT filter theoretically, it is not an altogether unreasonable scenario to assume for a real world SPIT filter. For example, one could assume that a *possible* relevant feature of calls is their *duration* (see Section 3). In this case SPIT calls will have a shorter duration than regular calls because after a callee answers the call, they will hang up as soon as they realize it is SPIT. The majority of regular calls on the other hand will tend to have a longer duration. While this certainly simplifies the situation from the real world, we can imagine that both durations can be modeled by an exponential distribution with an

average (expected) length of SPIT calls of $1/\lambda_0$ minutes and an average length of NON-SPIT calls of $1/\lambda_1$ minutes ($\frac{1}{\lambda_1} > \frac{1}{\lambda_0}$).

First, let us consider the expected stopping time from Eqs. (11)-(12). From Eqs. (13)-(14) we have that the Kullback-Leibler information number for Eq. (21) is given by

$$\begin{aligned}\kappa_0 &= \int_0^\infty \log\left[\frac{\lambda_1 \exp(-\lambda_1 x)}{\lambda_0 \exp(-\lambda_0 x)}\right] \cdot \lambda_0 \exp(-\lambda_0 x) dx \\ &= \log\frac{\lambda_1}{\lambda_0} \int_0^\infty \lambda_0 \exp(-\lambda_0 x) dx + (\lambda_0 - \lambda_1) \int_0^\infty x \cdot \lambda_0 \exp(-\lambda_0 x) dx \\ &= \log\frac{\lambda_1}{\lambda_0} + 1 - \frac{\lambda_1}{\lambda_0}. \end{aligned} \tag{22}$$

(On the second line, the first integral is an integral over a density and thus is equal to one; the second integral is the expectation of p_{SPIT} and thus is equal to $1/\lambda_0$.) Similarly we obtain for κ_1 the expression

$$\kappa_1 = \log\frac{\lambda_1}{\lambda_0} - 1 + \frac{\lambda_0}{\lambda_1}. \tag{23}$$

Note that for more complex forms of distributions we may no longer be able to evaluate κ_i in closed form.

As we can see, κ_i only depends on the ratio λ_1/λ_0. Thus for fixed accuracy parameters α, β the expected stopping time in Eqs. (11)-(12) will also only depend on the ratio λ_1/λ_0. The closer the ratio is to zero, the fewer samples will be needed (the problem becomes easier); the closer the ratio is to one, the more samples will be needed (the problem becomes harder). Of course this result is intuitively clear: the ratio λ_1/λ_0 determines how similar the distributions are.

In Table 1 we examine numerically the impact of the difficulty of the problem, in terms of the ratio λ_1/λ_0, on the expected number of samples until stopping for different settings of accuracy α, β. For instance, an average NON-SPIT call duration of 2 minutes as opposed to an average duration of SPIT calls of 12 s leads to $\lambda_1/\lambda_0 = 0.1$, and distributions that are sufficiently dissimilar to arrive with high accuracy at the correct decision within a very short observation horizon: with accuracy $\alpha, \beta = 0.001$, the filter has to observe on the average 1.0 calls if the source is NON-SPIT and 4.9 calls if the source is SPIT to make the correct decision in at least 99.9% of all cases. (Notice that the stopping time is not symmetric.)

Next we will compute the log-likelihood ratio Λ_t. From Eq. (3) we have

$$\begin{aligned}\Lambda_t &= \sum_{i=1}^t \log\frac{p(x_i|\text{NON-SPIT})}{p(x_i|\text{SPIT})} = \sum_{i=1}^t \log\frac{\lambda_1 \exp(-\lambda_1 x)}{\lambda_0 \exp(-\lambda_0 x)} \\ &= \sum_{i=1}^t \left[\log\frac{\lambda_1}{\lambda_0} + (\lambda_0 - \lambda_1) x_i\right].\end{aligned}$$

			$\alpha, \beta = 0.05$		$\alpha, \beta = 0.01$		$\alpha, \beta = 0.001$	
λ_1/λ_0	κ_0	κ_1	$E_{\text{SPIT}}[T]$	$E_{\text{NON}}[T]$	$E_{\text{SPIT}}[T]$	$E_{\text{NON}}[T]$	$E_{\text{SPIT}}[T]$	$E_{\text{NON}}[T]$
0.99	-0.00005	0.00005	52646.2	52294.7	89463.4	88865.9	136938.9	136024.5
0.95	-0.00129	0.00133	2049.0	1980.1	3481.9	3364.9	5329.7	5150.5
0.90	-0.00536	0.00575	494.3	460.8	840.0	783.0	1285.8	1198.6
0.70	-0.05667	0.07189	46.7	36.8	79.4	62.6	121.6	95.8
0.50	-0.19314	0.30685	13.7	8.6	23.3	14.6	35.6	22.4
0.30	-0.50397	1.12936	5.2	2.3	8.9	3.9	13.6	6.1
0.10	-1.40258	6.69741	1.8	0.3	3.2	0.6	4.9	1.0
0.01	-3.61517	94.39486	0.7	<0.1	1.2	<0.1	1.9	0.1

Table 1. How does the difficulty of the problem, expressed in terms of the ratio λ_1/λ_0, affect the expected number of samples until stopping, $E_{\text{SPIT}}[T]$ and $E_{\text{NON-SPIT}}[T]$, for different settings of the accuracy parameters α, β.

The decision regions for the SPRT from Eq. (4) are thus

$$\log \frac{\beta}{1-\alpha} < t \cdot \log \frac{\lambda_1}{\lambda_0} + (\lambda_0 - \lambda_1) \sum_{i=1}^{t} x_i < \log \frac{1-\beta}{\alpha} \tag{24}$$

or, equivalently,

$$\log \frac{\beta}{1-\alpha} + t \cdot \left(\log \frac{\lambda_0}{\lambda_1} \right) < (\lambda_0 - \lambda_1) \sum_{i=1}^{t} x_i < \log \frac{1-\beta}{\alpha} + t \cdot \left(\log \frac{\lambda_0}{\lambda_1} \right). \tag{25}$$

From the latter we can see that the boundaries of the decision regions are straight and parallel lines (as a function t of samples). Running the SPRT can now be graphically visualized as shown in Figure 1: the log-likelihood ratio Λ_T starts for $t = 1$ in the middle region between the decision boundaries and, with each new sample it observes from the unknown source, does a random walk over time. Eventually it will cross over one of the lines after which the corresponding decision is made. For a fixed value of α, β, changing the ratio λ_0/λ_1 changes the slope of the decision boundaries. For a fixed value of λ_0, λ_1, changing the accuracy α, β shifts the decision boundaries upward and downward.

3 Evaluation with real world data

To evaluate our approach with some real-world data, we used call logs from 106 subjects collected from mobile phones over several months by the MIT Media Lab and made publicly available in [2]. The dataset gives detailed information for each call and comprises about 100,000 regular voice calls. Ideally we would have liked to perform our experiments based on real-world data for both SPIT and NON-SPIT. Unfortunately, this dataset only contains information about

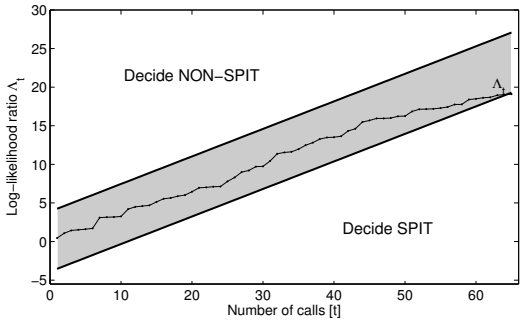

Fig. 1. An example run of SPRT. The log-likelihood ratio Λ_t starts at $t = 1$ in the region between the decision boundaries and, with each new sample observed from the unknown source, performs a random walk over time. Eventually it will cross over one of the decision boundaries and either enter the region marked "decide SPIT" or enter the region marked "decide NON-SPIT".

regular calls and not SPIT—and at the time of writing, no other such dataset for SPIT is publicly available.[1]

In the following we will take again *call duration* as feature for our filter to discriminate SPIT from NON-SPIT. To obtain call duration for SPIT calls, we proceed as follows. The dataset is artificially divided into two smaller datasets: one that corresponds to SPIT and one that corresponds to NON-SPIT. The set of SPIT calls is obtained by taking 20% of all calls whose call duration is <80 seconds, the remaining calls are assigned to the set of NON-SPIT calls.

Our experimental setup is the following: we first assign a source to be either of type SPIT or NON-SPIT. We then begin to draw samples uniformly at random from the matching data set and present the samples to the filter until it decides to either accept or block all future calls from this source. Since now we do not know the true generating distributions $p_{\text{SPIT}}, p_{\text{NON-SPIT}}$, we first fit via maximum likelihood an exponential distribution to the datasets we have built and later use the learned distributions as surrogate for the unknown distribution $p_{\text{SPIT}}, p_{\text{NON-SPIT}}$ in the SPIT filter (i.e., κ_1, κ_0 is calculated for the learned distributions which in our case have mean 30.23 seconds for SPIT and 129.64 seconds for NON-SPIT).

We performed 10,000,000 independent runs. For each setting, Table 2 shows the number of times the SPIT filter made the wrong decision and how many calls the filter needed to observe to arrive at this decision. These empirical quantities

[1] The earlier work described in [7] set out to precisely change that. In it the authors describe a methodology for creating SPIT traffic and also provide a common data set for the use in benchmark comparisons. However, the data set they provide is generated from "emulated users based on a social model"; in essence, the authors use common tools to generate the SPIT traffic, where the relevant features, such as call duration, inter-arrival time, behavior upon receiving a call, etc. are all modeled by sampling from distributions. For example, the call duration was generated from an exponential distribution the parameter of which was specified by hand (which amounts to the same as what we do here).

	source=NON-SPIT			source=SPIT		
α, β	Error	Stopping time	$E_{\text{NON-SPIT}}[T]$	Error	Stopping time	$E_{\text{SPIT}}[T]$
$1{\cdot}10^{-6}$	$6.16{\cdot}10^{-3}$	10.11	7.5	0	15.67	20.0
$1{\cdot}10^{-5}$	$1.39{\cdot}10^{-2}$	9.12	6.2	0	13.18	16.7
$1{\cdot}10^{-4}$	$3.15{\cdot}10^{-2}$	8.00	5.0	0	10.66	13.3
$1{\cdot}10^{-3}$	$6.96{\cdot}10^{-2}$	6.64	3.7	0	8.18	10.0
$1{\cdot}10^{-2}$	$1.54{\cdot}10^{-1}$	4.97	2.4	0	5.66	6.5
$1{\cdot}10^{-1}$	$3.40{\cdot}10^{-1}$	2.79	0.9	0	3.05	2.5

Table 2. Results of running the SPIT filter on real-world call data. For each of the two cases, i.e., source=NON-SPIT and source=SPIT, the table shows two types of quantities: the empirical ones obtained in simulation and the theoretical ones computed from the model. The first column, accuracy α and β, is the input parameter (values here chosen by hand) and gives an upper bound on the probability of making a mistake. The second column, error, then shows the empirical error rate. The third column, stopping time, shows the average number of observations before a decision was made. And the fourth column, $E[T]$ shows the corresponding expected stopping time from Eqs. (11) and (12). Note that the actual values do not always agree with the theoretical bounds: there is a mismatch between the true model and the distribution generating the data (see text).

are compared with the theoretical ones computed under the assumption that the model is correct, i.e., SPIT calls are drawn from an exponential distribution with $\lambda_0 = 1/30.23$ and NON-SPIT calls from an exponential distribution with $\lambda_1 = 1/129.64$. The results show that, with α, β for example set to 10^{-3}, the empirical error rate for NON-SPIT is $6.96 \cdot 10^{-2}$ (meaning that 6.96% of regular callers are wrongly identified as SPIT), while the empirical error rate for SPIT is 0 (meaning that 0% of SPIT bots are wrongly identified as regular users). The average number of calls the SPIT filter had to let through to arrive at this decision was 6.64 and 8.18, respectively. Note that while the error rate for NON-SPIT seems rather high (and is higher than the upper bound on the probability of making an error for this setting of accuracy, i.e., the value of α), we should keep in mind that in our experiment there is a mismatch between the true model and the distribution generating the data.

In practice, one would use more sophisticated (and more accurate) methods to estimate the distributions from data and, since the SPRT filter would ideally be just one component in the larger SPIT prevention framework and not be alone responsible for making the decision of whether to accept or reject the call, also allow higher tolerance thresholds for the error (which should be automatically adjusted as described in Section 2.3 by having a human operator define the cost of making an error and using optimization to find the best α^*, β^*).

4 Summary

In this paper, we presented the first theoretical approach to SPIT filtering that is based on a rigorous mathematical formulation of the underlying problem and, in consequence, allows one to derive performance guarantees in terms of worst case

cumulative misclassification cost (the expected loss) and thus, on the number of samples that are required to establish with the required level of confidence that a source is indeed a spitter. The method is optimal under the assumption of knowing the generating distributions, does not rely on manual tuning and tweaking of parameters, and is computationally simple and scalable. These are desirable features that make it a component of choice in a larger, autonomic framework.

Acknowledgements

Sylvain Martin (Post-Doctoral Researcher) acknowledges the financial support of the Belgian National Fund of Scientific Research (FNRS). Tobias Jung acknowledges financial support from a research fellowship of ULg. This work is also partially funded by EU project ResumeNet, FP7–224619.

References

1. N. Chaisamran, T. Okuda, G. Blanc, and S. Yamaguchi. Trust-based voip spam detection based on call duration and human relationships. In *Proc. of the 11th Int. Symp. on Applications and the Internet (SAINT)*, 2011.
2. N. Eagle, A. Pentland, and D. Lazer. Inferring social network structure using mobile phone data. *Proceedings of the National Academy of Sciences (PNAS)*, 106(36):15274–15278, 2009.
3. P. Kolan and R. Dantu. Socio-technical defense against voice spamming. In *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 2007.
4. M. Nassar, O. Dabbebi, R. Badonnel, and O. Festor. Risk management in voip infrastructure using support vector machines. In *International conference on Network and Service Management (CNSM'10)*, pages 48–55, 2010.
5. M. Nassar, S. Martin, G. Leduc, and O. Festor. Using decision trees for generating adaptive spit signatures. In *Proc. of the 4th International Conference on Security of Information and Networks (SIN 2011)*, 2011.
6. M. Nassar, R. State, and O. Festor. Monitoring sip traffic using support vector machines. In *Proc. of RAID '08*, pages 311–330, 2008.
7. M. Nassar, R. State, and O. Festor. Labeled VoIP data-set for intrusion detection evaluation. In *In: Proceedings of the 16th EUNICE/IFIP WG 6.6*, 2010.
8. J. Quittek, S. Niccolini, S. Tartarelli, M. Stiemerling, M. Brunner, and T. Ewald. Detecting SPIT calls by checking human communication patterns. In *IEEE International Conference on Communications (ICC 2007)*, June 2007.
9. R. Schlegel, S. Niccolini, S. Tartarelli, and M. Brunner. SPIT prevention framework. In *IEEE GLOBECOM'06*, pages 1–6, 2006.
10. D. Shin, J. Ahn, and C. Shim. Progressive multi gray-leveling: a voice spam protection algorithm. *IEEE Network*, 20:18–24, 2006.
11. Y. Soupionis, G. Tountas, and D. Gritzalis. Audio CAPTCHA for SIP-based VoIP. In *IFIP Adv. in Inf. & Com. Tech.*, volume 297, pages 25–38, 2009.
12. A. Wald. Sequential tests of statistical hypotheses. *Annals of Mathematical Statistics*, 16:117–186, 1945.
13. A. Wald and J. Wolfowitz. Optimum character of the sequential probability test. *Annals of Mathematical Statistics*, 19:326–339, 1948.

14. Y.-S. Wu, S. Bagchi, N. Singh, and R. Wita. Spam detection in voice-over-ip calls through semi-supervised clustering. In *Proceedings of the 2009 Dependable Systems Networks*, pages 307–316, 2009.
15. H. Yan, K. Sripanidkulchai, H. Zhang, Z.-Y. Shae, and D. Saha. Incorporating active fingerprinting into spit prevention systems. In *Third annual security workshop (VSW'06)*, 2006.