Simulation of SPIT filtering: Quantitative Evaluation of Parameter Tuning

F. Menna, R. Lo Cigno
Dip. Ingegneria e Scienza dell'Informazione
(DISI) – University of Trento, Italy

S. Niccolini, S. Tartarelli
NEC Europe Ltd. – Heidelberg, Germany

***Abstract*—A future where Internet Telephony will constitute a target valuable to attack is not so unrealistic. E-mail spam botnets software can be updated to send voice spam (commonly referred to as SPIT, SPam over Internet Telephony) constituting a huge threat to VoIP-based applications and business. This paper tries to learn from one of the biggest lessons learnt from the e-mail world, i.e. "do not wait until is too late", and proposes a quantitative study, based on a simulation campaign, of SPIT filtering based on the analysis of the call setup protocols. After discussing attack scenarios based on dichotomic choices by the attacker, it presents how the SPIT filtering system can be optimized in order to self-tune parameters achieving high SPIT detection ratio and low false rates at the same time.**

I. Introduction

E-mail spam, defined as the transmission of unsolicited mails, is one of the unsolved problems the Internet faces every day. Currently spam e-mails are estimated to exceed regular ones. One of the main reasons why this has happened was the lack of availability of solutions when the problem arose, together with the very low costs for sending e-mails. With the widespread adoption of IP Telephony systems allowing cheap (if not free) calls and the open connection with the Internet, a similar threat is expected to arise. The name of this threat is SPIT (SPam over Internet Telephony) and it refers to the transmission of unsolicited calls via Internet Telephony.

Similarly to e-mail spam, SPIT is regarded to be such a complex threat that there will not be a single method of protection. The most promising SPIT mitigation solution foreseeable is the combination of multiple approaches analyzing different characteristics of the call [1]. A good taxonomy and classification of SPIT mitigation techniques into different stages with increasing level of intrusiveness for the caller and the callee is reported in [2].

This paper considers non-intrusive SPIT detection mechanisms implemented in a simulator, in which results from different techniques are combined using a scoring mechanism like in [3]. The paper first aims at investigating the sensitivity of the SPIT filtering results to different attack scenarios. Based on the simulation results obtained, an innovative self-tuning mechanism is then proposed to automatically adapt to different attack scenarios.

The rest of the paper is organized as follows: Sect. I-A describes the related work and provides an understanding of the motivation and the positioning of the methodologies presented in this paper. Sect. II details the SPIT attack scenarios used in this work, while Sect. III reports the details of the modules' inter-working as well as details on their principles. Sect. IV presents quantitative results with a sensitivity study of the main parameters and the self-adaptation enhancements of the "statistical" module. Sect. V concludes the paper and reports on future work.

A. Related Work

The characteristics of e-mail spam today show that the main source of spam are botnets PCs[1]. One solution for mitigating attacks from this type of sources (which are most probably going to be re-used for sending SPIT) would be restricting the possibility of receiving communications only to known accounts ("white list" approach) that previously authenticated themselves with an introduction system. Such recommendation is reported in [4] together with an overview of building blocks for SPIT prevention that IETF suggests.

Clearly the "white list" approach violates the principles of communication where everybody is supposed to be able to contact everybody; additionally there are still unsolved issues related to how the user white lists should be automatically populated while keeping the false positive (good calls blocked as SPTI) ratio as low as possible.

An interesting work that tries to assess automated estimation of user trustworthiness using call duration, social networks and global reputation was proposed in [6]. Another approach to SPIT mitigation trying to identify trustworthy users is described in [5].

Recent evolutions of spam techniques have shown that spammers are starting to use compromised accounts from valid users (versus just having an unauthorized application running on botnets PCs). If the user identity can be stolen and their buddy lists used as destinations of session initiations then many of the techniques based on reputation systems and white lists become ineffective.

This suggests that mechanisms based on the identification of bad accounts ("black list" approach) are required to be combined with others based on the "white list" approach to offer

This work is supported in DISI by NEC Europe Heidelberg Laboratories under the contract *"SPIT Prevention Simulation (SPS)"*. Federico Menna is now working in Vodafone.

[1] Botnets PCs are large sets of compromised hosts running a remote-controlled malware able to generate spam messages.

978-1-4244-3435-0/09/$25.00 ©2009 IEEE

a broader mitigation effectiveness. Examples of mechanisms based on the identification of account to be blacklisted are reported in [7], [8], [9].

A known vulnerability of black list approaches currently not addressed in the literature is that spam sources have a volatile nature in terms of source addresses (IP addresses are often allocated dynamically to computers, e.g. home PCs connected to the networks of their ISP via ADSL or similar). Besides, blacklisting a compromises account prevents also its legitimate use.

This motivates the work object of this paper where black list mechanisms are given the possibility to send feedback to a black listing module while contributing to the overall estimation of blocking probability. This approach is able to achieve a black list population in a fast manner and keep it frequently updated.

In addition, this paper shows how SPIT detection accuracy can be improved by using self-tuning techniques and provides an extensive quantitative analysis of detection accuracy which is a novel contribution to the literature.

II. SPIT Attack Scenarios

How SPIT attacks to enterprise or carrier VoIP systems will be devised cannot be defined with certainty, because it will depend on the attackers' goals and the technicalities of the network operation. In order to evaluate possible counterattacks, however, it is necessary to assume an attacking model or scenario (because of the lack of real attack data as of today).

Starting from today spam goals and attack methodologies, we have devised different possible scenarios based on dichotomic choices by the attacker.

The first choice is between *hard* (or greedy) and *soft* attacks. Hard attacks try to make use of all possible resources for a short period of time, basing the strategy on a hit-and-run tactic, so that when the attack is identified and countermeasures are taken (such as legal actions and/or network obscuring), the attack is already over. Soft attacks are instead based on disguise, with the attacker trying to hide SPIT calls by spreading them randomly in space and time. Soft attacks can be carried out with botnet-like techniques, grabbing control of many machines that act independently but with coordination.

The second choice regards instead the use of spoofing. An attacker can simply use some 'random' IP address and SIP identity (No Spoofing – NoS), or may try to disguise himself by using some legitimate user identity (SpooFing – SpF).

Combining together these choices, we obtain four main scenarios: i) hard-NoS; ii) soft-NoS; iii) hard-SpF; and iv) soft-SpF. Many different flavors and variations of these four basic schemes can be found, and some of them are discussed while commenting quantitative results in Sect. IV.

III. Modules Roles and Inter-working

As already discussed in Sect. I, this work leverages the ideas and architecture presented in [1] and the simulator presented in [3]. One of the key concepts introduced in those works is the need for different *modules* to analyze different *characteristics* of the call setup and to blend together the results of the different modules in order to obtain high SPIT rejection ratio and low false rates at the same time. Here we focus on the optimization of modules devised for IP level and SIP syntax based filtering working during the call set-up phase. In particular, we test the efficiency of such mechanisms for a variety of attack scenarios and investigate how configuration parameters can impact the performance of the single modules.

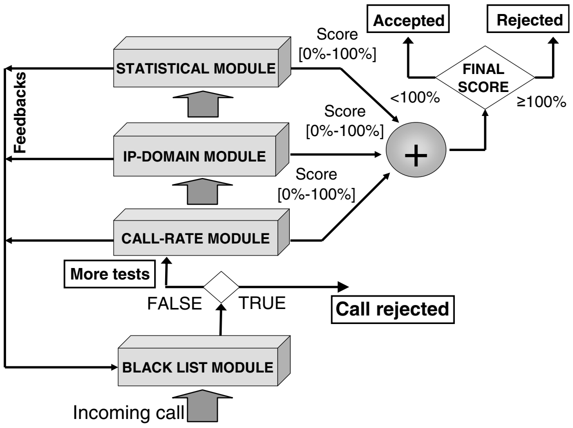

Fig. 1. SPIT filtering modules and their relations

Fig. 1 depicts the four modules that we have defined and tested as well as their relationship in making the final decision to accept or "reject" a call (note that for legal reason a call cannot be rejected by the system; therefore the word "rejection" is not used in a strict term in this paper, but rather means that the call is identified to be a SPIT one, and might be e.g. redirected to a voice-mail system). The scores of the modules are expressed as a percentage and summed up. Any call whose total score is 100% or more is "rejected" and does not reach the callee. A similar system was presented in [3], however, some major enhancements have been introduced for this paper.

The first main difference is related to the blacklisting module. In [3] blacklisting had a role similar to that of the other modules, so that its score was combined in a linear fashion to that of the other modules and a total score was evaluated. In this work, we decided to emphasize the role of blacklisting by first directing each incoming call to the blacklisting module. If the caller is blacklisted, then the call is directly "rejected" without being further processed by other modules. A key feature is that the blacklisting module is populated in a dynamic fashion, based on the feedback provided by the other modules (see description below).

A second major difference of the system studied in this paper compared to [3] is its capability of self-tuning configuration parameters. This feature will be discussed in detail in Sect. IV.

In the following we briefly describe the four modules object of our study.

Blacklist — The module filters IP addresses. All IP addresses observed for a given period of time are stored in a table together with a variable N_b, initially set to zero. When one of the other modules assigns *alone* a score 100% to a call, then it gives the blacklist a feedback and N_b is incremented by one. After a feedback from any module the identified IP address is blacklisted for a period of time $T_b = T_{\text{base}} \cdot N_b$.

When T_b expires the IP address is removed from the blacklist, but not deleted: N_b keeps its value and, if a new call coming from the IP address receives a score 100% from one of the modules, N_b is incremented by one, T_b is recomputed and the IP is blacklisted again.

This mechanism is self sustained and stable over a large range of T_{base}, and ensures that IP addresses blacklisted by mistake will quickly recover their normal status. In the simulations we finally set $T_{\text{base}} = 1\,\text{s}$, which is very fast and suitable for an operator network, where users cannot be prevented from accessing the service for long times. Larger values can be used in enterprise environments.

Call Rate — The rationale behind this module is that a normal caller will not place calls with high rates. Thus given an observation interval τ_o (sliding window), the module assigns the following score based on the number of observed calls N_o within τ_o:

$$S_{cr} = 100 \cdot \min\left(1, \left[\max\left(0, \frac{N_o - Th_1}{Th_2 - Th_1}\right)\right]\right)$$

i.e., a linearly increasing score between the two configuration thresholds Th_1 and Th_2 (calls per minute), which are parameters available for tuning. In order to identify calls from the same caller it is necessary to consider a reliable identifier. We decided to look at the IP signalling address in the 'INVITE' method, since this parameter needs to be true for the caller to be able to receive the '200 OK' ack.

IP-domain — This module is devoted to classify and filter attacks based on spoofing. There are three different possible "suspicious" situations:

A Different usernames with the same domain and IP address: a situation that may arise from an enterprise proxy, but also from an attack; the level of suspicion (or base score) BS_A is low;

B The same SIP identity associated with different IP addresses: a situation that may arise from IP spoofing, but also from highly mobile terminals; the level of suspicion BS_B is medium;

C Different domains associated with the same IP address: the situation is highly unlikely in normal networks so the level of suspecion BS_C is high.

The total base score assigned is computed as

$$\text{BS}_{IP} = M_A \cdot \text{BS}_A + M_B \cdot \text{BS}_B + M_C \cdot \text{BS}_C$$

where M_A, M_B, and M_C are the numbers of calls in the previous observation interval that share the situation A, B or C with the call currently analyzed.

To derive the final score, this module requires a training phase, where a fingerprint of good traffic is taken in form of an histogram $H(\text{BS}_{IP})$ containing the distribution of the base scores for good calls. After the training, the score of a call is assigned complementing the histogram normalized to its maximum value and multiplying by a weighting factor C_f. Let $\hat{H}$ be the value of the most probable point (mode) of the histogram, then the score of this module is

$$S_{IP} = C_f \cdot \left(\frac{H(\text{BS}_{IP})}{\hat{H}}\right)$$

BS_A, BS_B, BS_C, and C_f are all parameters subject to optimization.

Statistical — The last module is devoted to identify regularities in the call arrival distribution. The rational for the statistical module is that an artificial SPIT-call generator will normally try to exploit resources by placing calls as soon as resources are available. The key idea is therefore looking for inter-times measured as the time elapsed since when the last call was hung up and the time the next call started. The SPIT generator might try to minimize this time, so that resources are mostly exploited. In this case, the inter-time defined above will be mainly due to network delays that, at least in the short term, we can assume to follow a stationary distribution. The SPIT generator might however be more sophisticated and add some randomization in the generation of calls. Even in this case, such randomization would follow some statistical rule, which would be captured by the statistical module. The details of how statistical analysis is performed can be found in [10]. The evaluated statistical inference is normalized to 1 and transformed into a score by multiplying it by a base score BS_{st}, which is again a tuning parameter.

The final score depends also on the average number of contemporaneous estimations had until that moment (AV) and it is computed with the following formula:

$$S_{st} = \begin{cases} 0 & AV > 20 \\ (B) \cdot (1 - \frac{AV}{20}) \cdot (BS_{st}) & AV \leq 20 \end{cases}$$

where B is a binary score assigned by the module (0 for a good call, 1 for a bad one)

The module includes also a mechanism that switches the module off when the number of active estimations is too high ($AV > 20$).

IV. Quantitative Results

In the following we present selected results from different attack scenarios, tuning the different parameters and showing that it is impossible to find values of the parameters fitting any scenario. Based on this observation we finally propose a closed-loop control to automatically drive some parameters to their optimal value depending on the attack scenario.

The good calls are generated based on a standard telephone call model assuming Poisson arrivals and exponential holding times (5-minutes average, accounting for the fact that VoIP is normally used by heavy telephone users who place longer calls). However, from the SIP-VoIP perspective, the traffic scenario is rather sophisticated. We took into account the different actors currently playing the VoIP business, starting from service providers to virtual service provider to large and small enterprises, taking into account the possibility of having SIP proxies and how all or this affects the traffic patterns and protocol parameters (from the codec rate to the domains distribution). The attacks are generated considering *greedy* attackers that try to place a new call as soon as the previous one is terminated. In order to take into account the network delay, a uniformly distributed inter-time (between the end of a call and the beginning of the next one) is introduced, varying between 50 ms and 250 ms. Attackers' calls are assumed to be significantly shorter than good calls, with an average duration of 15 seconds. The attacks last for the entire simulation. Finally, we set the capacity of the link on which the anti-SPIT is run to 100 Mbit/s, so that, with a normal mix of codecs the traffic intensity is around 100 k-Erlangs, corresponding to a collection point of a medium-large provider. In the following all time measures are in seconds and call rates in calls per minute (CpM).

A. Sensitivity analysis on the main parameters

The parameters analyzed in this paper are:

- the upper threshold (Th_2) of the Call Rate module;
- the complementing factor (C_f) of the IP-Domain module;
- the score (BS$_{st}$) of the Statistical module.

Indeed, in [10] we analyzed all parameters, but space forbids to report here all results, so we selected those that have more impact on the performance. If not otherwise stated, all base scores (and in particular BS$_{st}$) are set to 100%. Only in some special cases they need to be changed and this is discussed case-by-case.

1) Hard-NoS: in this scenario the attack is carried out by 10 attackers and each of them can handle 10 simultaneous calls. In this case the main role is played by the Call Rate module: the analyzed parameter is Th_2. Fig. 2 shows how the false positives (good calls blocked) percentage gets lower as the threshold Th_2 increases, while the false negatives (bad calls not recognized) percentage remains almost stable. The other modules don't help in reducing false positives and negatives ratios. Fig. 2 clearly depicts that the best value for Th_2 is 16 CpM: the final false positives and false negatives percentages are respectively 0.01% and 0.18%.

2) Soft-NoS: this scenario represents a situation where one spitter places only one call at a time in order to avoid the Call Rate module and, since the attack is not spoofed, even the IP domain module doesn't help in revealing the spit calls: the only module that can succeed in finding out the bad calls is the statistical one, in the assumption that the spitter will not try to emulate a standard user behavior because this has a too low usage of resources. Fig. 3 shows that the statistical module must be able to assign a score (BS$_{st}$) higher than 100% to the incoming calls in order to block bad calls. In particular BS$_{st}$ $= 115$ is enough to reduce false negatives close to zero, obtaining a false positives percentage of 0.58% and a false negatives percentage of 0.31%. The reason lies in the fact that the module is based on the statistical inference of an inter-time not being drawn from an exponential distribution. As such, it only gives a probabilistic answer which is smaller than one.

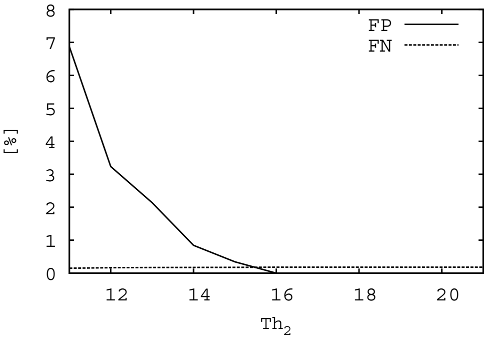

Fig. 2. Hard-NoS attack: False Positives/Negatives vs Th_2

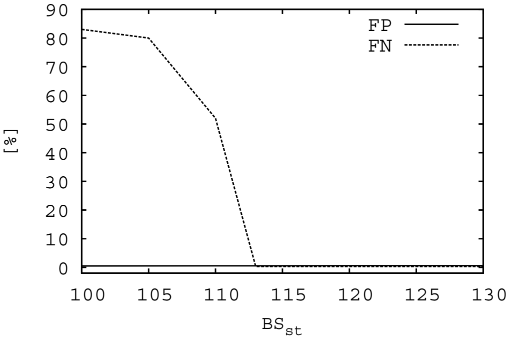

Fig. 3. Soft-NoS attack: False Positives/Negatives vs BS$_{st}$

3) Hard-SpF: in this scenario the spoofing technique is used by one spitter together with a high attack rate (the attacker can handle up to 200 simultaneous calls), so the main role in detecting bad calls is played by the Call Rate and IP Domain modules. For the Call Rate module we propose a configuration like the one used to fight hard-NoS attacks (Th_2=16), while for the IP Domain module we analyzed the effect of varying the C_f parameter as presented in Fig. 4. The C_f value that minimizes the false positives percentage without causing an excessive growth of the false negatives one is 20. The results obtained in this scenario with the proposed configuration for the two modules are 0.01% of false positives and 0.04% of false negatives.

4) Soft-SpF: this kind of attack is characterized by one spitter, using the spoofing technique together with a low call rate (up to 10 simultaneous calls). A good cooperation between the Statistical and the IP Domain module is needed in order to efficiently reveal this kind of attack. So the crucial parameters are C_f and BS$_{st}$. The configuration proposed for soft-NoS (BS$_{st}$=115) and hard-SpF (C_f=20) attacks leads to 0.17% of false positives and 0.75% of false negatives.

The results obtained in the four scenarios, and summarized in table I, are very good in terms of both false positives and false negatives percentages, but, while for Th_2 and C_f it is possible

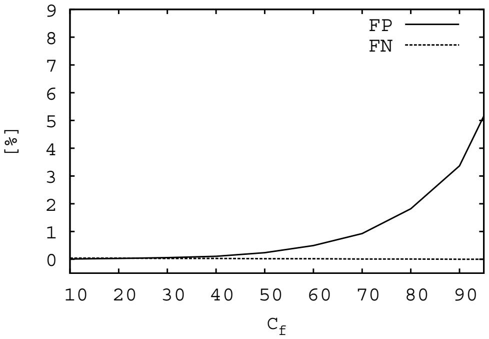

Fig. 4. False Positives/Negatives vs C_f

ATTACK	FP	FN
Hard-NoS	0.01%	0.18%
Soft-NoS	0.58%	0.31%
Hard-SpF	0.01%	0.04%
Soft-SpF	0.17%	0.75%

TABLE I
SUMMARY OF FALSE INDICATION RESULTS

to find values (16 and 20, respectively) that are suitable for every scenario, for BS$_{st}$ it is not possible to find a value valid for all cases: soft attacks need a value of 115% in order to efficiently reveal a high number of bad calls while, at the same time, in hard attacks the module must use a value not higher than 100% to avoid an increase of false positives (see Fig.5).

Sect. IV-B deals with this problem and proposes an adaptive solution.

B. Self adaptation and tracking of the statistical module

The sensitivity analysis presented in Sect. IV-A highlighted the difficulty in finding a value for BS$_{st}$ that fits well different attack scenarios, while all other parameters seem to have values that yields good results for a broad range of situations.

A naïve solution can be trying to infer from the incoming traffic the type of attack and set the optimal BS$_{st}$ accordingly. However, this solution looks simplistic and cumbersome at the same time. Additionally, it cannot adapt to "new" scenarios by definition.

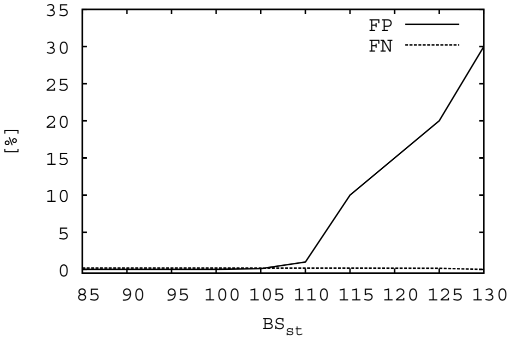

Fig. 5. Hard attacks: False Positives/Negatives vs BS_{st}

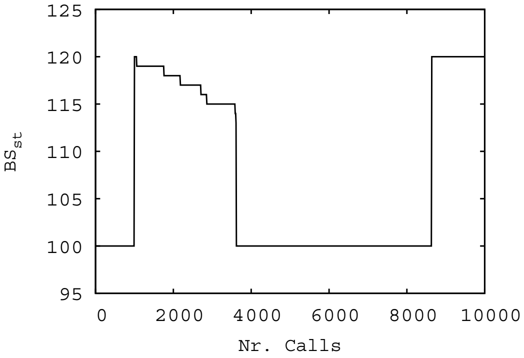

Fig. 6. Mixed attacks: BS_{st} vs Nr. Calls

Analyzing the results in Sect. IV-A, it is clear that higher BS$_{st}$ leads to more false positives, while lower BS$_{st}$ leads to higher false negative rates; and this independently from the attack scenario. This observation lead us to define a very simple positional, step-wise controller that minimizes the H_∞ norm of a weighted sum of the false positive ratio R_{fp} and the false negative ratio R_{fn}[2].

The step-wise, discrete time controller is defined as

$$BS_{st}(n+1) = BS_{st}(n)\left[1+\alpha R_{fp}(n)+\beta R_{fn}(n)\right]$$

where n is the discrete time. The sampling is triggered by an observation interval of 100 calls, thus it is not constant in time.

In [10] we analyzed the two proportional parameters, showing that there is a directly proportional relationship between BS$_{st}$ and the false positives percentage and an inversely proportional relationship between BS$_{st}$ and the false negatives percentage, thus the two proportional parameters must be: $\alpha < 0$ and $\beta > 0$.

In order to better evaluate the effectiveness of the implemented automatic tracking system, we present some test where we mixed the various kind of attacks during the same simulation: for example, we considered a situation where the first 2000 calls follow a soft-NoS scenario, then for about 1000 calls there is a mix between a hard-NoS and soft-SpF attack and after further 750 calls the attacks switches to a hard-NoS for approximately 2500 when it suddenly switches back to a soft-NoS situation until the end of the simulation. This situation is represented in Fig.6 that clearly depicts how the BS$_{st}$ parameter reacts very well to attacks' variations.

Other results obtained in mixed scenarios are summarized in table II, which clearly shows the effectiveness of the automatic tracking mechanism in all the proposed mixed scenarios.

The presented mechanism is a pure proportional control loop system since BS$_{st}$ depends only on the measured values

[2] The H_{infty} norm measures the total amount of a given quantity (e.g., absolute difference w.r.t. a reference level) as the time $t \rightsquigarrow \infty$. In our case the quantity is the ratio of falses, weighted to keep into account that false positives are much more annoying than false negatives

	SELF CONF.		**STD. CONF.**	
Attacks	**FP**	**FN**	**FP**	**FN**
Hard-NoS+Soft-NoS	0.65%	1.22%	1.16%	3.21%
Hard-SpF+Soft-NoS	0.21%	0.66%	0.21%	1.94%
Soft-SpF+Soft-NoS	0.58%	2.55%	0.37%	6.59%

TABLE II
SUMMARY OF THE RESULTS WITH MIXED ATTACK SCENARIOS WITH AND WITHOUT TRACKING

of false positives/negatives. In order to improve its effectiveness, the introduction of a derivative and an integrative part in the adaptive function can be studied, but this is left for future work together with the extension of the automatic parameters setting to other components of the system.

V. Discussion and Conclusions

With the adoption of IP Telephony systems, a threat similar to e-mail spam is expected to arise, with unsolicited calls, carrying entirely or partially registered calls being aggressively placed toward randomly selected users. To counter SPIT (SPam over IP Telephony) a few proposals have been already published, presenting means how to hypothetically address such a (not yet) widespread threat. This paper, building on previous results, addressed issues related to parameters tuning and quantitative analysis of several filtering modules that can be cooperatively used to identify STIP calls. The results show that opportune design and combination of detection methodologies analyzing different *characteristics* of the call setup can be tuned to obtain high SPIT rejection ratio and low false alarms (both positive and negative) rates at the same time. A very innovative contribution of this paper is the achievement of a configuration able to defend users from SPIT in a wide range of possible attack scenarios thanks to proper parameters' tuning and self-adaptation improvements of one of the key modules.

Object of further work will be the use of real traffic as simulation input in terms of good calls as well as an additional effort in the modeling of bad calls and attack scenarios.

References

[1] R. Schlegel, S. Niccolini, S. Tartarelli, M. Brunner, "SPam over Internet Telephony (SPIT) Prevention Framework," In Proc. IEEE GLOBECOM '06, S. Francisco, CA, USA, Nov. 27 – Dec. 1, 2006.

[2] J. Quittek, S. Niccolini, S. Tartarelli, R. Schlegel, "On SPam over Internet Telephony (SPIT) Prevention" *IEEE Communication Magazine*, Vol, 46, Issue 8, pp. 80–86, Aug. 2008.

[3] M. Falomi, R. Garroppo, S. Niccolini, "Simulation and Optimization of SPIT Detection Frameworks" In Proc. IEEE GLOBECOM '07, Washington, DC, USA, Nov. 26 – Nov. 30, 2007.

[4] J. Rosenberg, C. Jennings, "The Session Initiation Protocol (SIP) and Spam" IETF RFC 5039, Jan. 2008.

[5] R. Dantu, P. Kolan, "Detecting Spam in VoIP Networks" In Proc. SRUTI '05, Cambridge, MA, USA, Jul. 7 – Jul. 8, 2005.

[6] V. Balasubramaniyan, M. Ahamad, H. Park "CallRank: Combating SPIT Using Call Duration, Social Networks and Global Reputation" In Proc. CEAS '07, Mountain View, CA, USA, Aug. 2 – Aug. 3, 2007.

[7] D. Waiting, N. Ventura, "A Multilayered Architecture for Preventing Automated Spam in the IP Multimedia Subsystem" In Proc. IEEE GLOBECOM '07, Washington, DC, USA, Nov. 26 – Nov. 30, 2007.

[8] A. Salehin, N. Ventura, "Blocking Unsolicited Voice Calls Using Decoys for the IMS" In Proc. IEEE ICC '07, Glasgow, Ireland, Jun. 24 – Jun. 28, 2007.

[9] H. Kang, Z. Zhang, S. Ranjan,A. Nucci, "SIP-based VoIP Traffic Behavior Profiling and Its Applications" In Proc. ACM MINENET '07, San Diego, CA, USA, Jun. 12, 2007.

[10] F. Menna, "Spam Over Internet Telephony Prevention Systems: Design and Enhancements with a Simulative Approach," MSc Thesis, Università degli Studi di Trento, Italy, 2007. Available on-line at http://disi.unitn.it/locigno/tesi