Project no.:	**COOP-32720**
Project full title:	**SPam over Internet telephony Detection sERvice**
Project Acronym:	**SPIDER**
Deliverable no.:	**D2.3**
Title of the deliverable:	**General anti-spam security framework for VoIP Infrastructures**

Contractual Date of Delivery to the CEC:	31.07.2007
Author(s):	Y. Rebahi, S. Ehlert, S. Dritsas, G. F. Marias, D. Gritzalis, B. Pannier, O. Capsada, T. Golubenco, J. F. Juell, M. Hoffmann
Participant(s):	Fokus, AU, VozTelecom, Iptego, Eleven, Telio
Work package contributing to the deliverable:	WP2
Dissemination level:	Private
Version:	0.9
Total number of pages:	30

Abstract:
Based on the results achieved in the previous deliverables D2.1 and D2.2, where respectively a spit threat analysis was performed and an overview of the current anti-spit techniques was provided, the spider architecture is designed. In fact, a special care is dedicated to extracting the major issues behind the persistence of spam. Accordingly, some methods and algorithms are suggested for building the spider framework. The Spider architecture has adopted a modular vision. This allows the proposed solutions to be updated, upgraded and adjusted to the needs of the providers without any difficulty. As a first step, different intelligent modules are specified. Some of them are common, however, they are redefined here because they will certainly play an important role in the fight against spit in the future. Moreover, the spider project will implement them in a more sophisticated way taking into account the VoIP specificities. Some other modules employing new concepts such as reputation and audio analysis are also considered in spider, which shows the creativity and innovation within this project. With other respects, some instances of the generic architecture are provided. Indeed, for each identified use case, a high level description of the overall architecture, the behavior of the involved modules as well as the used interfaces are given.

Keyword list:
VoIP, SIP, SPAM, SPIT, Fake identities, Open relays, Open proxies, Modular architecture, Generic architecture, White/black lists, Challenge/response, RFC4474, Audio analysis, Reputation, Open relays, Use cases

Table of contents

1 Introduction

In the current deliverable, we aim at providing a high-level architecture description for the spider anti-spit solutions. Based on the results achieved in the previous deliverables D2.1 and D2.2, where respectively a spit threat analysis was performed and an overview of the current anti-spit techniques was provided, the spider architecture is designed. In fact, a special care is dedicated to extracting the major issues behind the persistence of spam. Accordingly, some methods and algorithms are suggested for building the spider framework.

The Spider architecture has adopted a modular vision. This allows the proposed solutions to be updated, upgraded and adjusted to the needs of the providers without any difficulty. This generic architecture identifies two main layers. A detection layer where the spit detection intelligence is distributed over different modules and a decision layer where the results received from the detection layer are gathered and accordingly a decision is made.

As a first step, different intelligent modules are specified. In fact, some common techniques such as white/black lists and Human Interactive Proofs are redefined here because we think that these techniques will certainly play an important role in the fight against spit in the future. However, the spider project will implement these techniques in a more sophisticated way taking into account the VoIP specificities. In addition to that, some other modules employing new concepts such as reputation and audio analysis are also considered which reflects the creativity and innovation within this project.

After introducing the detection intelligence through the mentioned modules, some instances of the generic architecture are provided. The reason is to show that more than one solution exist and a selection between the intelligent modules is possible according to the providers' requirements and preferences. Indeed, for each identified use case, a high level, but detailed, description of the overall architecture, the behavior of the involved modules as well as the used interfaces are given.

2 Overview of architecture

2.1 SPIT and Denial of Service (DoS) attacks

Given the traction Voice over IP has received the past years, traditional security concerns have been mitigated within SIP. These concerns are slowly being directed towards pre-emptive awareness, recently manifested by *SANS Top 20 Internet Security Attack Targets*[1]. This section limits itself to one area of concern, discussing briefly the difference between SPIT and Denial of Service ("DoS").

SPIT is not the unique threat of the current VoIP infrastructures. DoS attacks are another one. Although both have a negative impact on the VoIP networks, SPIT and DoS attacks express completely different goals and objectives. SPIT, aims at passing some sort of information (related to money, medicine, jobs, etc) to the recipients without their consent. However, DoS attacks are generated with the intent of disturbing legitimate traffic, potentially rendering any given network-based service useless. As SPIT is also sent (or generated) in bulk, it might be possible that this huge volume of SPIT might reach a level where it becomes some kind of denial of service attacks.

Although SPIT has not yet been dispersed in professional environments with financial and marketing motives, we will expect (in the future) to see variations of the classical DoS headache. Most notably will be the early media phenomenon.

As depicted in Figure 1, an early media DoS occurrence will be divided into two destinations: Through domain proxy (or rogue proxy) and the direct target to end-users (local domain policies will apply). Sending high volume of pre-recorded messages to either of the destinations will result in high bandwidth consumption, choking network resources, and then expeditiously causing loss of service.

[1] [Link] - http://www.sans.org/top20/

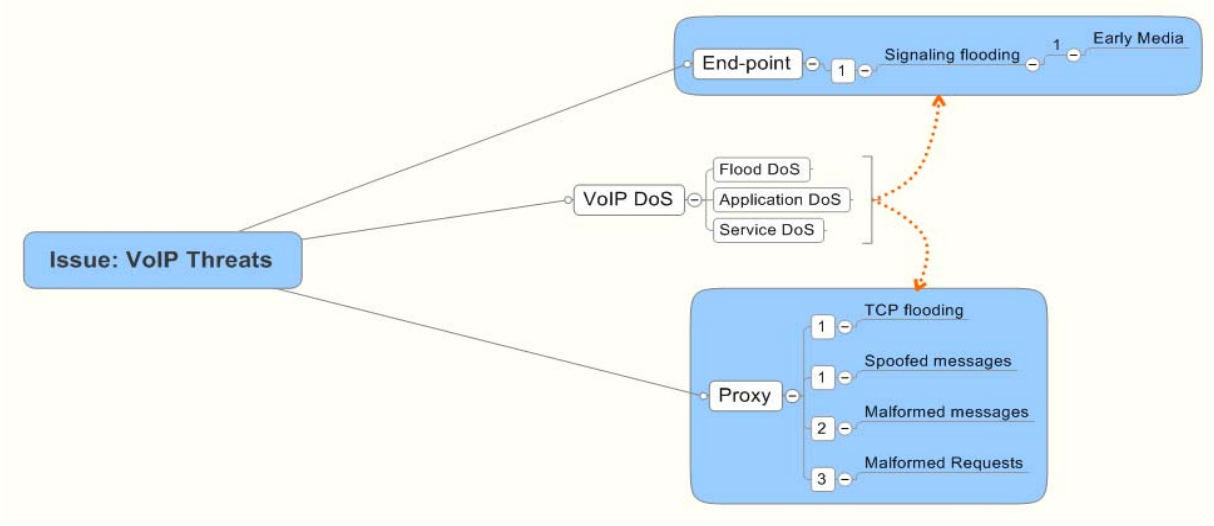

Figure 1: Early Media techniques

By utilizing trends and experiences from today's e-mail situation, it is easy to draw the conclusion that viruses will see its growth through not-yet existing exploitations. In the interest of reaching the mass-marked, drone-networks might be assumed to start flourishing, thus validating the aforementioned high volume of pre-recorded messages.

Lastly, we will also see SIP transactions being manually crafted into templates, and then used for automated outbound calls. If we make the assumption that not all these transactions are correctly implemented according to RFC 3261 [RFC3261] and applicable extensions, CPU cycles in end-points or proxies might experience a considerable increase, again causing loss of service.

2.2 Intra and inter domain spit handling

The trajectory for Voice over IP versus PSTN usage is expected to reach in the 90 percentile by 2009 in favor of VoIP, and by that implying the need for a greater, more robust VoIP infrastructure. The original VoIP pretext since inception was to function according to the open island model from the e-mail world. With the rise of new islands all over the world, there is now a general consensus for a need of Federations[2] to create and sustain peering between domains. Certain prerequisites must be clearly defined and strictly upheld for this to be effective. This section will, on a high level, cover the most common ways to limit SPIT incidents within own domain and between direct peering domains.

Having a strong local domain infrastructure is a priori principle for controlling and maintaining end-points, and verifying the validity of these. Enforcing a strong identity [RFC4474] policy will provide the building blocks, consequently paving the way for added security means between Member Islands ("MI") [SCH06]. Provider locations, compatibility between providers, transport methods, trust relationships, and defined routing are some of the vital placeholders. Now we have established communication with the means to identify end-point users in own domains as well as from the external peering MI's.

[2] Federation: Cluster of peering ITSP's

The SPIDER framework has outlined a set of consecutive modules, which will iterate the preceding one, and a selection of these will play the bigger part in performing the actual inter-domain SPIT recognition patterns. MI's must come to a level of agreement on how stringent the domain-traversal should be, and a solution would be to share whitelists and blacklists. Each domain will maintain their own global white-/blacklist, and these are compared when a call is initiated from external Islands. This will be easily extended to other domain policies, and can be weighted according to trust relationships.

2.3 Overview of the SPIT solutions evaluation

In order to fight SPIT phenomenon several anti-SPIT mechanisms have been proposed that adopt techniques used in the email spam paradigm. Furthermore, some of them appear to be the basic blocks-modules of more sophisticated anti-SPIT architectures that have been proposed in the literature. The most important techniques, nowadays, are: black, white and gray lists, content filtering, challenge-response schemes, consent-based approaches, reputation-based mechanisms, SIP addresses management and charging-based mechanisms. Hence, in the next paragraphs, we present the most important anti-SPIT architectures that have been proposed so far with an eye towards their applicability and feasibility for fighting SPIT.

2.3.1 Anti-SPIT mechanisms

2.3.1.1 Anonymous Verifying Authorities

This technique, presented in [CRO05], is based on a "call-me-back" scheme and the usage of two entities, namely: (a) the Mediator and (b) the Anonymous Verifying Authority (AVA). The proposed mechanism ties to mitigate SPIT by anonymously blocking unwanted calls through AVA and the Mediator. Thus, in the case of not call establishment, the caller is not aware of the existence of the callee.

2.3.1.2 Network-Level Anti-Spit Entity

A network-level entity, placed in the edge of the network, is proposed in [MAT06]. The role of this entity is to analyze the transmitted SIP packets, and to detect SPIT according to specific detection criteria. By using these criteria, a weighed sum is introduced, namely *spitLevel*, which functions as a threshold. If the *spitLevel* is exceeded specific actions are performed depending on the policies adopted by the callee's domain.

2.3.1.3 Reputation and Charging

The work in [REB06] proposes two techniques for handling SPIT. The first is based on reputation builds trust within different SIP communities and uses the resulting trust networks for detecting SIP spam. The second is a variant of the payment at risk proposal.

2.3.1.4 DAPES

In DAPES (Domain-based Authentication and Policy-Enforced for SIP), any SIP-based communication passes through two stages of verification; namely, verification of the caller's identity, and mutual authentication of the participated proxies alongside with verification of the outbound proxy [SRI04].

2.3.1.5 Progressive Multi Gray-Levelling (PGM)

The PMG, proposed in [DON06], stems from the anti-SPAM graylisting concept. For that reason, it calculates and assigns a non permanent gray level for each caller, in order to check if a message is SPIT or not. This level is calculated based on previous call patterns of a particular caller. Depending on the level's value, appropriate actions are taken.

2.3.1.6 Biometric countermeasure

In [BAU06], the authors propose the use of global servers that bind users' identities to personal data; they select biometric data, such as a person's voice. The proposal is based on the concept of binding identities to persons that cannot change globally.

2.3.1.7 RFC 4474

An end-user authentication scheme is discussed in RFC 4474 [RFC4474], based on Public Key Infrastructures (PKI) and Certificate Authorities (CA). Although this approach is not oriented specifically for SPIT handling, the identity control mechanism is useful for controlling SPIT. For identity control, two new SIP header fields have been proposed.

2.3.1.8 SIP SAML

The Security Assertion Markup Language (SAML) is used for the expression of security assertions, such as authentication, role membership, or permissions. In [TSC06], a new approach using SAML for SIP authentication and authorization through asserted traits is proposed. The authors aim at a strict identity control accomplishment, in order to prevent spiters change their identity frequently.

2.3.1.9 Differentiated SIP

In [MAD06], an extension to SIP is proposed, called Differentiated SIP (DSIP). It tries to handle SPIT through the classification of callers into three categories of lists, namely: white, who are legitimate callers, black, who are spitters, and grey list, who are not yet recognized-classified. Through this classification the handling of calls is conducted according. When the caller is unknown, a human verification test is imposed, in order to prove that she is not a bot machine.

2.3.1.10 VoIP Seal

This concept, presented in [NICO], is a two-stage system: The first stage has invisible modules and the second interacting ones. Each module of the first stage contributes a score in [-1,1], where high score correspond to a high probability that the call is SPIT. Each module is associated with a weight, and the final score is compared with two thresholds. If the score is higher than the lower threshold, then the call passes to the second stage modules, which could be a Turing test; otherwise the call is rejected.

2.3.1.11 Voice Spam Detector

The Voice Spam Detector (VSD) [DAN05] combines many anti-SPIT approaches. The system is a multi-stage SPIT filter based on trust and reputation, and uses feedbacks between the different stages. Presence filtering, the first step, depends on the current status of the callee. The next step, that is the traffic pattern filter, analyzes the rate of incoming calls. This step is followed by the white/black lists' filtering. Bayesian filtering is the fourth step, where a call is checked regarding the behavior of the participated entities. Finally, reputation techniques are

used to check the acceptance of a call based on social relationships that an end- user maintains with other entities in the VoIP environment.

2.3.2 Evaluation

In order to evaluate current anti-SPIT mechanisms and techniques we have chosen several quantitative and qualitative criteria [MAR07]. Based on these criteria, we argue that the effectiveness and applicability of the proposed anti-SPIT mechanisms is judged as not adequate. In the next paragraphs we present the identified criteria while in Table 1 we depict the compliance of the proposed anti-SPIT techniques to these criteria.

- ***Percentage of SPIT calls avoided.*** How many SPIT call attempts have been identified by the anti-SPIT mechanism?
- ***Reliability.*** The precision of making the right adjustments about SPIT calls and callers, in terms of false positive and negative rates.
- ***Promptitude.*** Due to the real-time nature of VoIP, quick decisions regarding SPIT detection are a major requirement, especially when legitimate calls are analyzed.
- ***Human Interference.*** This metric represents the transparency of the anti-SPIT mechanisms to the end-user.
- ***Resources Overhead for the SIP provider.*** SIP providers should estimate the required resources for the implementation of the mechanism.
- ***Vulnerabilities.*** This parameter refers to the capability of a spitter to bypass any of the anti-spit countermeasures.
- ***Privacy Risk.*** This criterion is associated with the collection, manipulation, and dissemination of private data.
- ***Scalability***. This is an important criterion, since VoIP networks grow fast. Scalability should be considered when authentication is involved [RFC4474], since PKI and CA might need to establish complex cross-certification chains, or reputations and assertions are used [REB06].

A set of additional criteria that might apply to all proposed mechanisms include:

- ***Adoption:*** This parameter corresponds to the success of the anti-SPIT countermeasure, and depends on the effort it takes for an end-user, or a provider, to begin using it.
- ***Availability***. It denotes the increase in the availability of network, computing, memory, or human resources when preventing, detecting or handling spit countermeasures apply.

Criteria / Anti-SPIT Mechanisms	Percentage of spit calls avoided	Reliability	Promptitude	Human Interference	Resources Overhead for the SIP provider	Vulnerabilities	Privacy Risk	Scalability	Adoption	Availability
AVA	–	–	–	–	–	–	–	–	–	–
Anti-Spit Entity	☑	☑	☑	–	☑	–	–	–	–	–
Reputation/Charging		–	–	–	–	–	–	–	–	–
DAPES	–	–	–	–	–	☑	–	–	–	–
PMG	☑	–	☑	–	–	☑	–	–	–	–

Biometrics	–	–	–	☑	–	☑	–	–	–	–
RFC 4474	–	–	–	–	–	☑	☑	–	–	–
SIP SAML	–	–	–	☑	–	☑	☑	–	–	–
DSIP	–	–	–	–	–	☑	–	–	–	–
VoIP Seal	–	–	–	☑	–	–	–	–	–	–
VSD	☑	☑	–	☑	–	–	–	☑	–	–

Table 1: Compliance of anti-SPIT mechanisms to qualitative and quantitative criteria

The above table presents those SPIT mechanisms that measure, estimate or just mention the aforementioned criteria. It must be pointed out that the results of our analysis have not been based on which criteria each mechanism could, should, can or may fulfill, but we rather provide the existence of each criterion in the mechanisms' description.
For instance, in the description of VoIP Seal mechanism [NICO], it is mentioned that the particular mechanism is intrusive for the end user, as there is an interactive part requiring user's feedback. In our analysis, we do not estimate how intrusive this mechanism is or not, but we emphasize that there are information in the description of the mechanism that could help someone value accordingly the particular criterion, namely Human Interference. Finally, regarding the vulnerabilities criterion we must emphasize that we only focus on the vulnerabilities considerations that the authors take into account, whether these are few or many.

2.3.3 Innefficiency, Complexity and Dependency

Taking into account the aforementioned table, one can conclude that the current anti-SPIT mechanisms are somehow inefficient to address the majority of evaluation criteria. More specifically, most of them try to satisfy only some straightforward criteria of anti-SPIT identification and handling, while they ignore more sophisticated ways of handle SPIT such as the usability, and the overall adoption parameters. Finally, most of the proposed mechanisms are quite complex and use several building blocks that are adopted from the email anti-spam paradigm, thus making the overall applicability more difficult in a real VoIP environment.

2.4 Major current issues in SPIT

2.4.1 Protocol/software vulnerabilities

Voice over IP (VoIP) is the technology which supports voice and instant messages communications by exploiting the Internet infrastructure in conjunction with protocols such as SIP [SIP3261], RTP [RTP1889], etc. Besides the obvious advantages of VoIP, the use of the Internet as the underlying infrastructure might introduce many threats and vulnerabilities. Several of these threats and vulnerabilities are related to mail spam phenomenon, which in the case of VoIP context has been identified as Spam over Internet Telephony (SPIT) [ROS06]. In the next section the most important threats and vulnerabilities regarding SPIT phenomenon is presented.

Generally speaking, the majority of computer attacks usually take place through the exploitation of a set of vulnerabilities. Malicious users typically follow certain methodologies, often employing serially interconnected attacks in order to achieve their goal [MEH06]. In the spam context, for example, a corresponding methodology is to first gather a list of mail

addresses, then to prepare the spam message and, at the end forward the message to the intended recipients. Therefore, an in depth analysis of the techniques and methods used by attackers is a critical step for the protection of a system or a network. Predicting attackers' future attacks, together with the vulnerable points, which are identified after an attack, can be used for the prevention of similar future attacks, as well as for designing and implementing a reliable and adequate countermeasure to handle them.

In this context, and regarding the SPIT phenomenon we argue that it is important to comprehensively study the SIP protocol vulnerabilities in order to better handle and counter SPIT attacks. More specifically, in order to better implement a reliable and effective anti-SPIT mechanism we should recognize all the SPIT related SIP threats and vulnerabilities.

So, the threats and vulnerabilities that we have identified are classified into four categories, namely: (a) threats due to SIP protocol vulnerabilities, (b) threats due to the SIP protocol optional recommendations, (c) threats due to interoperability with other protocols, and (d) threats due to other (generic) security risks [DRI07].

The next table summarizes the most important threats and vulnerabilities,

Threats Category	Description	Recognized Threats
Threats due to SIP protocol vulnerabilities	Exploit a mandatory characteristic in the description of the structure and functionality of the SIP protocol.	✓Sending ambiguous requests to proxies ✓Listening to a multicast address ✓Population of "active" addresses. ✓Contacting a redirect server with ambiguous requests. ✓Misuse of stateless servers ✓Anonymous SIP servers and B2BUAs. ✓Sending messages to multicast addresses ✓Exploitation of forking proxies ✓Exploitation of messages and header fields
SIP's Optional Recommendations	Exploit optional recommendations of the SIP protocol.	✓Exploitation of registrars servers
Interoperability with other protocols	Exploit weaknesses of protocols used by SIP	✓Exploitation of particular domains' address resolution procedures
Other Security Risks	Exploit security weaknesses	✓Monitoring traffic near SIP servers ✓Port scanning on well-known SIP ports ✓Proxy-in-the-middle ✓Exploitation of the record-route header field

Table 2: SIP threats and vulnerabilities

2.4.2 Fake identities and anonymity

One of the main reasons behind the persistence of the spam activities is the difficulty to determine the origins of calls and messages. This can be achieved by using spoofed IP addresses or anonymity services.

The identity claimed by a spammer as well as the claimed domain might be some fake data or represent some information related to another domain that has nothing to do with spam. One of the most difficult problems in dealing with spam is the inability to determine accurately who sent the message (or the call) and how it was routed in the Internet. Without a strong authentication mechanism, these lists can harm more than help in spam detection especially that entire domains may be blocked even if they are not guilty.

The VoIP technologies also allow the users to utilize aliases and anonymity services. Anonymity is a way of protecting private data such as SIP URI and phone number. When anonymity is employed, the recipient is unable to identify the origin of the incoming calls. This service that should be used for protecting the private users data, can unfortunately be used for carrying out spit activities. This can be achieved simply by using the anonymity service when generating spit requests.

2.4.3 Insecure Third-party relays

A way for spiters to hide their identities is to take advantage of SIP open relay servers. A SIP open relay is a SIP proxy that accepts unauthorized third party relays of SIP messages even they are not destined to its domain. These relays could be used by spiters to route large volumes of unwanted SIP messages without being detected.

SIP open proxies are another kind of third-party relay, however, they are not specifically used for SIP, they could also route some other protocols requests.

Spiters are able to locate accessible third-party SIP servers by using free available automated tools. By relaying instant messages or calls through several open relay SIP proxies at the same time, it is possible to flood the SIP network with large amounts of spit calls in a very short time before being detected.

2.4.4 Automated SPIT

These are equipments or software that generate calls and send messages automatically. Any SIP client that can be configured to generate a huge number of simultaneously calls can be used for this purpose. The SIP testing tool SIPp can be used as an example of these automatic generators. These automatic tools may be used alone or with other equipment such as Interactive Voice Response (IVR) to deliver pre-recorded voice messages to the chosen recipients. Figure 2 describes a scenario where the spam information consists of a recorded audio file that will be played once the recipient picks the phone to answer. One can note that the call session in this case is initiated by a third party. In SIP, this functionality could be provided by the REFER method [RFC3515] if it is supported by the recipient’s terminal.

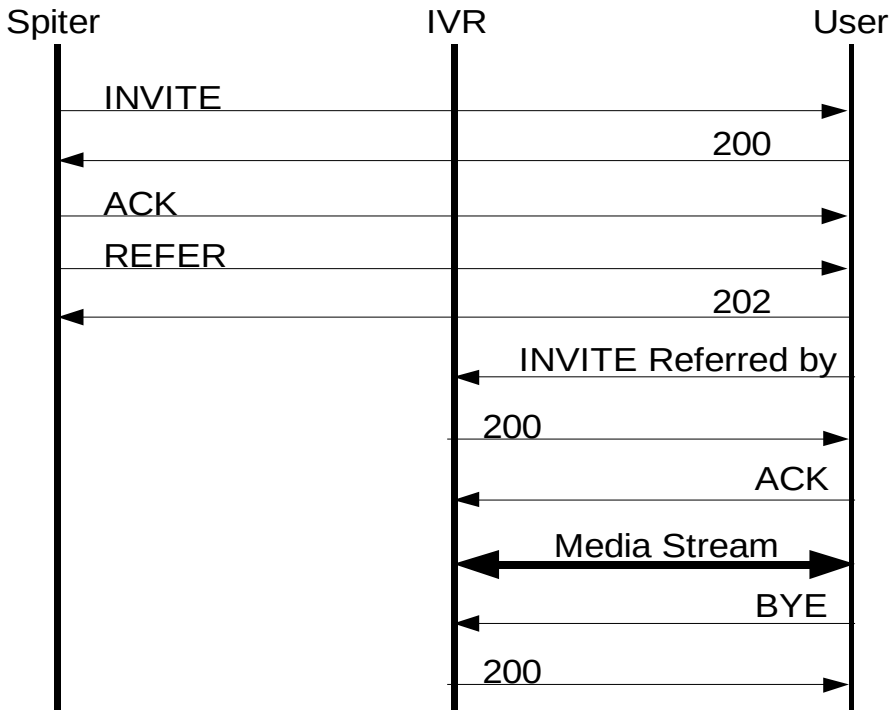

Figure 2: SPIT generated by automats

Let us mention that an efficient way to deal with the spit automat problem is to use the challenge/response mechanism or the audio analyzer, which are going to be discussed later on.

2.4.5 Human beings spiters

Telemarketing is a way for advertising products and offer services. It is common that companies and organizations use professional telemarketers or call centers to make phone calls and send messages to potential customers on their behalf. The phone calls here could be PSTN or VoIP based. We mentioned in the previous deliverables (D2.2 and D2.1) that PSTN was the main tool used by spammers to carry out their activities and Voice over IP (VoIP) was the potential carrier for spam in the future because of the convergence of voice and data provided by this technology. Although, this spam category (generated by human beings) is also annoying and frightening, its impact is very limited if we compare it with the impact of the spam generated by automata. On the other side, the use of white and black lists or reputation-based techniques can be an efficient way to mitigate this kind of spit activities.

3 Architectural subsystems and components

This section provides an overview of the anti-spit architecture that SPIDER intends to develop. The objective is to give a general idea of what is each architectural component for, and the functionalities it provides. As for the details, they will be provided in the next deliverable (i.e D3.1).

3.1 Generic architecture description

The objective of this document is to define a common architectural view for the anti-spit solutions. This generic vision follows a modular approach and hence will ensure the following,

- incorporation of newly detection mechanisms in response to the emergence of new SPIT forms activities
- seamless replacement of spit detection mechanisms without harming the overall architecture
- flexibility of the suggested solution to be adjusted to the providers requirements. This means that for a provider whose customers do not want to be challenged, this provider can use the audio analyser (or another technique) as a substitute.

Therefore, our design approach lies in the introduction of two basic layers,

- *Detection layer:* the main task of this layer is to test the received SIP requests against some predefined rules describing the spam behaviour. This layer consists of one module or several ones that operate in general in a separate fashion. The results of these tests is passed to the Decision point which will integrate the received results from the different modules and decide whether the SIP request (or the call) under consideration is spit or not. Some concrete examples of modules will be discussed in section 3.1.1.
- *Decision layer:* the main task of the Decision Point is to receive the results of the tests performed by the detection modules -- as mentioned earlier -- , combine them, and classify the intercepted requests as spit or not. The Decision Point could be either an independent component or just a functionality collocated with another component in our architecture. One way for combining the test results assumes that a score is assigned to each test. By combining these scores together in an appropriate way, a real number is generated and compared to a pre-defined threshold. If this number is greater than the threshold, the SIP request is classified as spit. In the opposite case, the request is classified as non spit. Another way of combining the tests results is to use them in a sequence. In other words, if the detection modules are organized in the sequence, and if the test corresponding to the first module is negative, a next test is performed, otherwise, the received request is classified as spit. For this kind of combination, a simple script can be used. A third way would be to build a decision tree in dependency of which modules exists. The tree could be described in a scripting language where the scripting engine traversals the tree, calls every Anti Spit module until a final decision is made. This solution adds more flexibility how to handle the results of each module and opens the ability of interaction between the modules. But this also adds more complexity to the whole system and makes it more difficult to add or remove modules to the system.

The architecture we are describing is depicted in Figure 3. The SIP server is the component that intercepts the SIP requests and passes the corresponding information to the Decision Point which in turn triggers the spit testing and takes a decision according to the corresponding results. The decision information will be passed back to the SIP server which is going to forward further the request or to reject it.

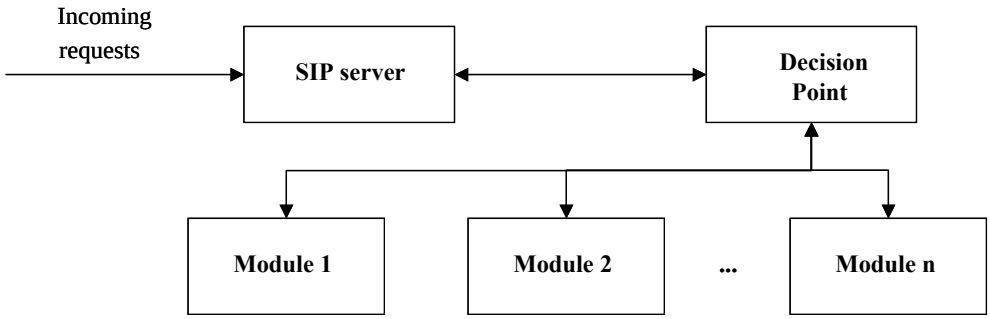

Figure 3: Anti-spit generic architecture

3.1.1 SPIT module examples

Here, we provide a nonexclusive list of spit detection modules that can be used for building potential anti-spit solutions.

3.1.1.1 Authentication module

One of the major reasons behind the persistence of spam is the use of spoofed identities. In such activities, receiving responses to the sent messages or to the established calls is not the main target. As a consequence, packets with spoofed addresses are suitable for such attacks especially that they are more difficult to filter since each spoofed packet appears to come from a different address and, thus, the true identity of the spammer is kept hidden.

The authentication module simply aims at enhancing domain verification for the anti-spit solutions. This module tries to identify senders by the set of SIP servers they use to send their requests. The idea behind the domain verification techniques is to get some information from the sender's domain that helps to verify that this domain is exactly the source of the message. As a consequence, a SIP message issued, apparently, from iptel.org could not be sent by a user outside iptel.org. The solution adopted here is described in RFC 4474 [RFC4474]. We assume in this technique that there is a trusted third party that is able to issue certificates, which could be in particular acquired by the SIP proxies in the network. To be more concrete, this technique works as follows,

- Before being able to establish a call, the caller needs to authenticate himself against the SIP proxy of his domain
- If the authentication succeeds, the SIP proxy in the caller domain computes a hash (using its private key) on some header fields of the SIP message and inserts the result as a new header field (called, *Identity header*) in the same SIP message. The SIP proxy also creates a second header field (called, *Identity-info*) and inserts the URL of the certification authority in it. After that, the SIP message is forwarded further.
- When the SIP proxy on the callee side, receives the SIP message, it uses the URL within the "Identity-info" field to retrieve the public key used the SIP proxy of the originating domain. This key is used to verify the signature that the originating SIP proxy has inserted in the forwarded SIP message.

3.1.1.2 White/Black lists module

These lists were already discussed in details in the deliverable D2.2. A spit black list is a list of IP addresses and domains of known spit users and servers. In practice, black lists are used to block all SIP requests coming from certain SIP servers identified as being used to send spam.

White lists are the opposite of blacklists. They involve trusted IP addresses and domains that are always allowed to establish calls and send instant messages, no matter what the content is. White lists allow calls and Instant messages coming from a trusted provider to come through, however, they do not provide a solution for blocking spit. These lists require constant maintenance to be very effective. If they are not properly maintained, the risk of losing legitimate calls and Instant Messages is high.

The white/black lists can be simply setup by allowing the users or the administrators to set lists either of users that they are allowed to contact the list owner or the users (or domains) that have to be blocked. In fact there are different ways for setting up these lists, some of them will be discussed in more details in the deliverable D3.1.

3.1.1.3 Machine learning module

This module can be used to automatically filter spit based on some machine learning techniques.The SIP message can be considered as a bloc of text to which a learning algorithm (decision tree, probabilistic approach, etc) could be applied. Because of the similarities between email and SIP instant messaging, this technique can be applied successfully to detect spit instant messages. As for SIP calls, one can restrict himself to the SIP message header because of the very useful data that it involves.

The SIP message header contains related information to the « From » field, the « To » field, the « Subject » field and some other data that might be valuable for detecting spit. The learning algorithm, when applied to the SIP message header, may deal particularly with the following situations,

- The *From* field

%yacine@domain_name.com	URI not valid : starts with %
1237@hotmail.com	URI not valid : starts with numbers. Pay attention to the PSTN case
yacine@yahoo	URI not valid : short domain name
Fokus.fraunhofer.de	URI not valid : no @
@yahoo.com	URI not valid : too short

- The *Subject* field : Is this field empty. If not, which words are there : pills, money, or some other good indicators ?
- Is this request the first one from this user ?
- Number of blank lines in the SIP header

As spit is often sent in bulk, a big number of the recipients URIs or IP addresses come from old lists which are no longer in operation. Also, spitters might use « dictionary attack » or trying out local parts « after the @ » in order to build a list of their targets. As a consequence,

the SIP messages sent by a spitter will present various weaknesses that a learning algorithm can exploit to detect the corresponding spit activity.

With other respects, the above criteria are defined according to our observation of the current email case. The learning algorithms need concrete data about the SIP messages sent by the spitters. Unfortunately, this is still early because the spit problem is expected to be more serious, hence detectable, as far as the VoIP deployment goes further. In any case, we intend within the SPIDER project to design just a basic machine learning module that could be elaborated afterwards when more spit data is available.

3.1.1.4 Proxy check module

This module will be in charge of checking whether the issued calls are using open relay servers (section 2.4.3). If any suspicious information is present in the SIP message header, this information is checked against a list reporting available open relay proxies. If this information matches any entry in the list, the call will simply be dropped. The mentioned list can be built through communities efforts. This simply means that the SIP administrators can cooperate in order to build a blacklist of open relay servers that can be used for blocking any SIP requests coming from these servers. One way for checking for open relays is to use automated tools. If we take the email case as an example, one can see such communities efforts in the web site (*www.stayinvisible.com/index.pl/proxy_list*) (see Figure 4). As for the available tools for checking open proxies and relay servers, one can check the following software (*http://yaph.sourceforge.net*).

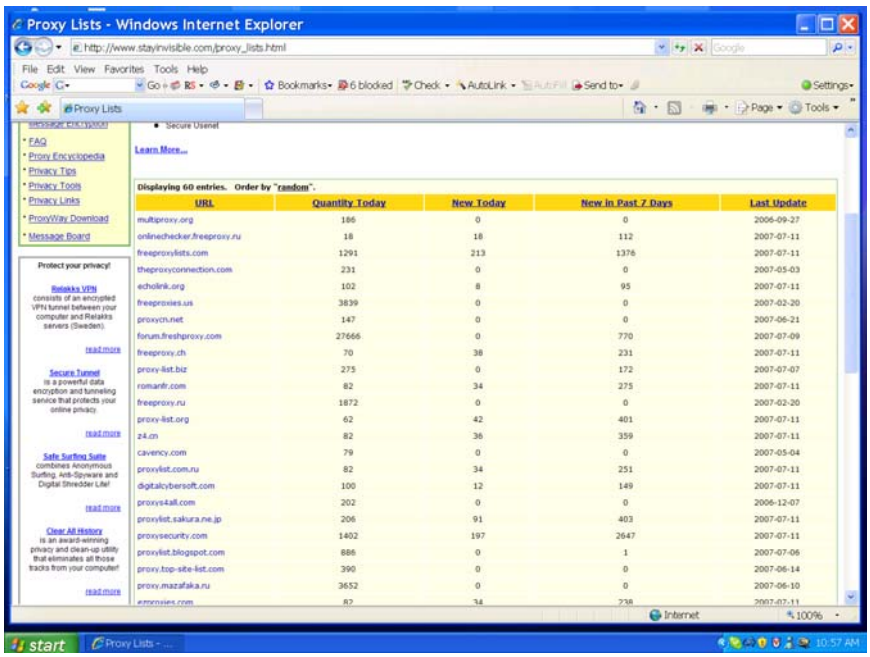

Figure 4: An example of open proxies in email

3.1.1.5 Challenge/Response module

Challenge/Response (C/R) is a common authentication / registration technique whereby an entity is prompted to provide a valid answer or some information in order to have access to

computational resources. A well-known challenge-response protocol is user authentication via pass phrase, where a service provider is asking for a pass phrase ("challenge") and the valid answer ("response") is the correct password. Since C/R is widely used to prevent SPAM, it can be used to prevent SPIT, as well.

The idea behind a C/R spam preventing system is that the spammer is not willing to consume time or resources to respond to a challenge, while people who are interested in contacting with the callee will be willing to spend a minute or part of their resources.

There are two main C/R categories that can be considered for the anti-SPIT scenario. The first one is about economic solutions, so called as computational puzzles. These puzzles prompt the SIP client to perform a calculation that limits the rate of incoming spit calls and increases their establishment cost. The second one is about puzzles, which can tell humans and computers apart. These are puzzles, which are considered easy for a human and really hard for an automated process to solve.

3.1.1.6 Reputation Manager

The techniques mentioned above within this section, although they are common, continue to be very useful in the fight against spam in general and will probably remain so for the future. The reputation concept is one of the newest means used in this fight. Indeed, if we check the email case, most of the anti spam products or solutions implements some sort of reputation. In the literature, only few reputation-based anti spit solutions can be reported. What make these solutions different is their ways of using the reputation concept in this context. To show how reputation can be used for fighting spit, we adopt here the technique described in [REB06]. This technique can be used in conjunction with one or more of the above-mentioned anti-spit methods in order to provide more complete solutions.

The reputation Manager is one of the major components described in the architecture depicted in Figure 5 (for more details, we refer to [REB06]). This scheme uses the "amount" of the chain trust that may exist between the SIP request receiver and the SIP request sender. In fact, each SIP subscriber is supposed to score the persons on his contact list according to the trust that he has in them. The amount of trust is expressed by a real number that we assume to be between 0 and 1 for simplicity. The greater this value, the more trustful the scored person is. When combining all these rated contact lists, a weighted graph is generated as depicted in Figure 5.

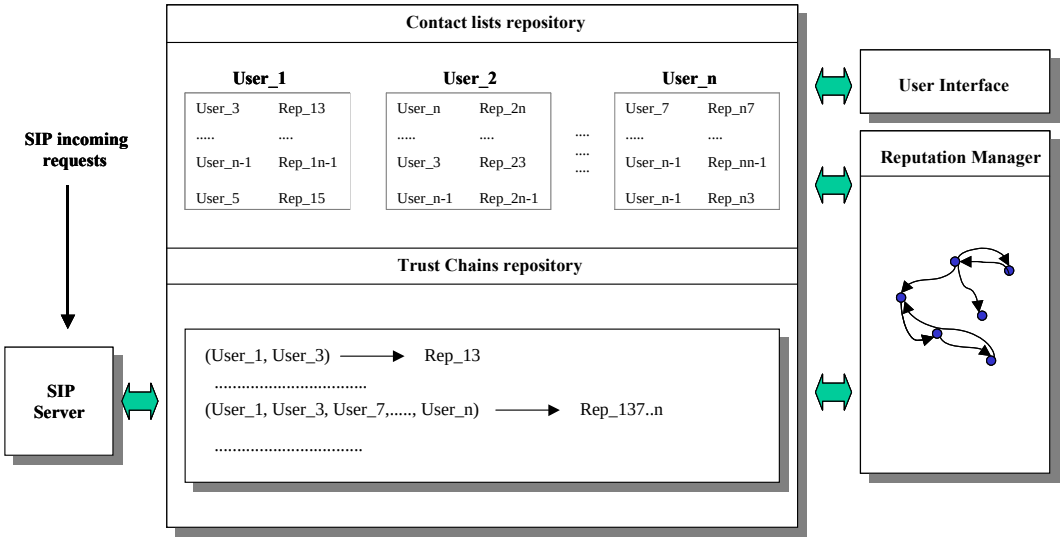

Figure 5: A reputation-based architecture

In addition to that, the Reputation Manager generates all possible paths between any two given SIP users. These paths are assigned a reputation value according to some appropriate metrics defined in [REB06] and are stored in the trust chains repository. This operation is regularly performed in order to minimize the SIP proxy response time to the incoming SIP requests.

If a path between the source and the destination is found, the corresponding computed score is compared to a predefined threshold. Based on the comparison result, the SIP request is accepted or rejected.

3.1.1.7 Audio analyser

The audio analyzer module is intended to be used for fighting spit generated by automats. This implies that the audio message which is heard by the receiver is always the same for the same spit wave. So a calculated checksum of the audio data, which are sent by the Spitter is the same. In a normal VoIP communication the audio checksums of two different calls are never the same. If a checksum is seen more than once the probability of Spit is very high for this signature (so also for the call).

The audio signature must be calculated before the call reaches the receiver. For that reason, all calls from unknown callers will be handled by an Audio Analyzer. The Audio Analyzer will receive the call, calculate the checksum and redirect the call to the receiver if the audio checksum was not seen before.

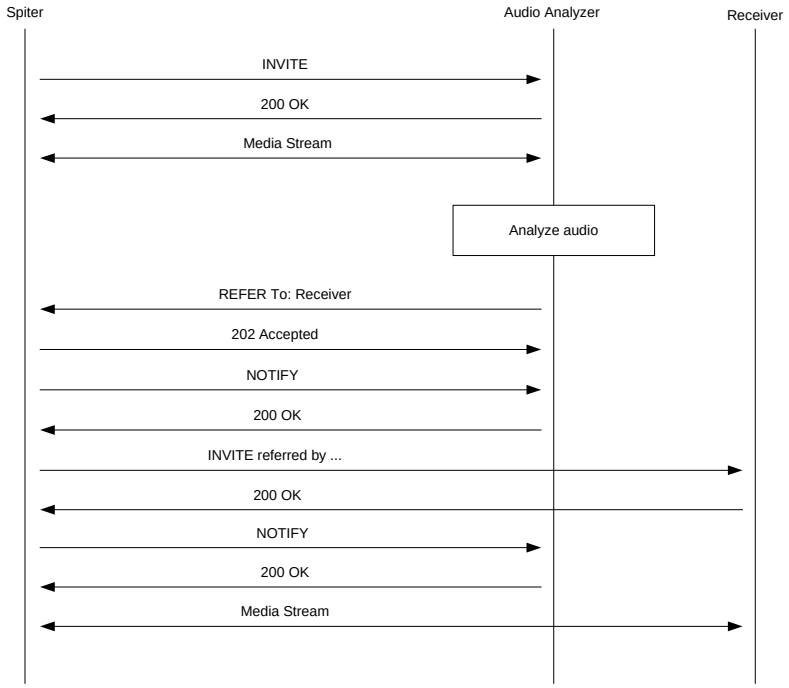

Figure 6: A message flow for the audio analysis technique

The Analyzer has similarities with an Interactive Voice Response (IVR). It answers a call, plays a prerecorded audio for human caller. This audio informs the caller of a short delay while processing his call. This does not only inform human callers, it also simulates a human receiver for Spiters, which are waiting for an audio answer before sending their messages. While playing the information the Audio Analyzer starts recording the caller audio. The recorded audio data will be feed to a checksum calculating algorithm, the checksum is checked against a database. If the counter of the checksum is unknown this call is potentially no spit. Otherwise the call is very likely to be spit.

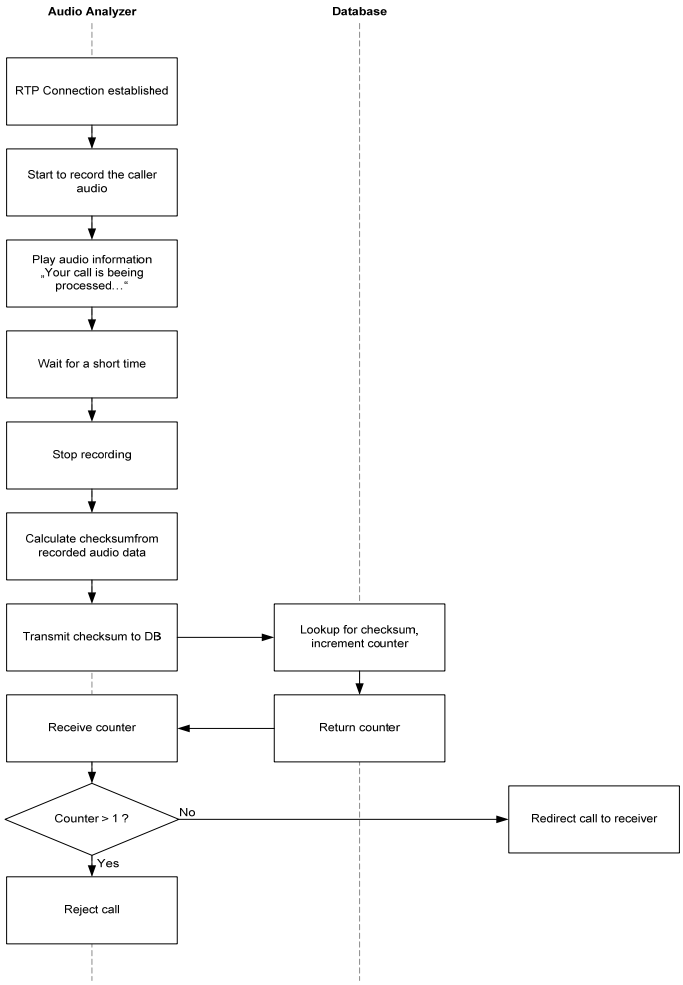

Figure 7: The audio analysis technique description

3.2 SPIDER architecture use cases

In section 3.1, we have suggested a generic architecture to face the spit problem. In this chapter, however, we provide some use cases (that are not exclusive) just to show how are architecture works in a practical way. In fact, different combinations of the detection modules are investigated and accordingly, some more complete anti-spit solutions are built.

3.2.1 Use case 1 : Authenticated challenge/response solution

The current use case is depicted in Figure 8. The involved components have already been discussed in section 3.1.1 and are not going to be discussed here again. We will restrict ourselves only to the description of how the corresponding architecture is used for filtering spit.

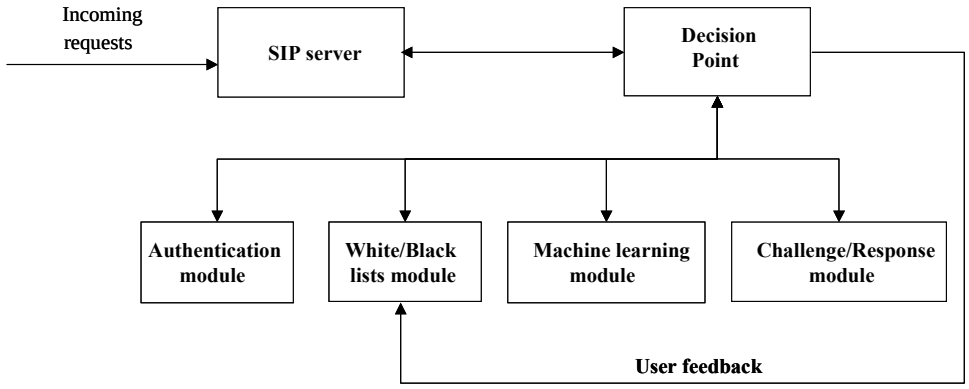

Figure 8: Authentication challenge/response architecture

Upon receiving a SIP request, the SIP proxy forwards this request to the Decision Point. The latter, in turn, forwards it further to the first detection module in this solution, namely, the authentication module. The latter will assure that the caller is exactly who is claiming to be, which means that the caller is not using a spoofed IP address or a fake domain name. If the operation triggered by this module (see section 3.1.1.1) fails, the call will be dropped. If this operation succeeds or if one of the domains (inbound or outbound) is not implementing this technique, the call will be treated as a normal SIP transaction, so it will be submitted to the next anti-spit test, which is represented by the white/black lists module. This module analyses the corresponding URI information and compare it to the entries stored in the black lists repository. If the request comes from a domain listed in the black list, the request will be rejected without any further processing. This means that the white/black list module will inform the Decision Point about that, which in turn informs the SIP server about the decision. As a consequence, the SIP server will simply reject the request under consideration. The SIP requests might be also originated through cable modems and DSL lines, this requires that the corresponding domains have to be taken into account as well.

If the domain from which the SIP request was issued is not on the black list, a second test is performed. Here, the domain information of the intercepted SIP request is checked against the information stored in the White list. If the request comes from a domain listed in the white list, the request will be forwarded without any further processing. In this case, the white/black list module will inform the Decision Point about that, which in turn informs the SIP server about the decision. The SIP server will simply process the considered request.

If the caller is not on the white/black lists, the request is passed to another test. The latter, called machine learning module, will examines the content of the SIP message and some other related information as describe in section…In fact, this module will help in distinguishing between potential spit requests and potential legitimate calls, so only potential spiters will be challenges using the challenge/response (C/R) module.

Our approach for a C/R SPIT prevention mechanism is based on a basic hypothesis: the caller is authenticated by her/his SIP provider. Furthermore, our mechanism uses reverse Turing tests (CAPTCHAs) and not computational puzzles based on hash functions. This is because

the VoIP communication should be able to be established between low CPU power devices (e.g., mobile or handheld phones) and the delay of solving hash-based computational puzzles, may be unacceptable by a legitimate caller.

If the machine learning module shows that the intercepted request could come from a spiter, a CAPTCHA challenge will be initiated by the callee's domain and sent to the caller. The latter will send his response and based on its correctness, the call will be dropped or challenged again. This will be discussed in more details in the deliverable D3.1.

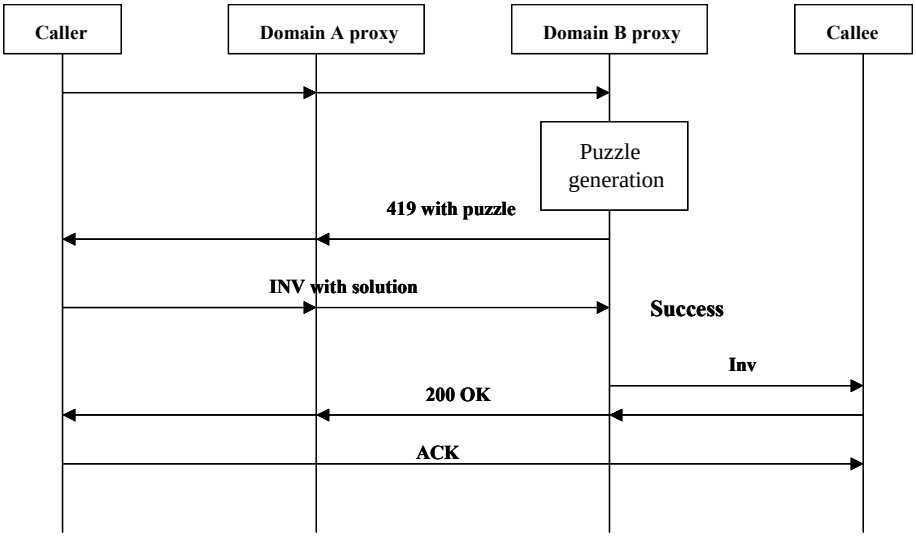

Figure 9: The case of a successful C/R test

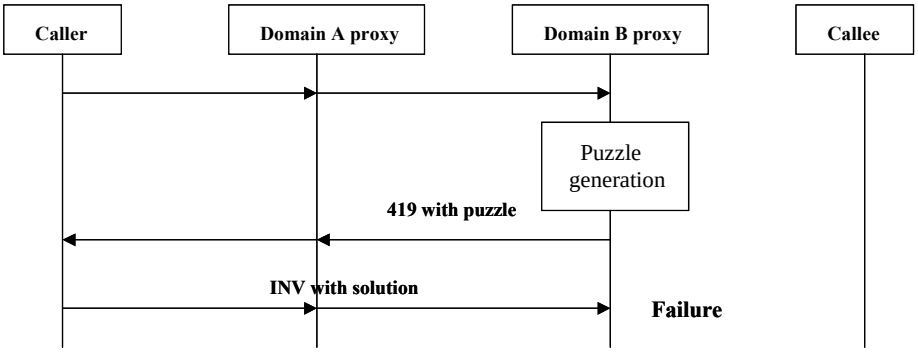

Figure 10: The case of a failed C/R test

The main advantage of the challenge/response mechanism is that it can utilize any CAPTCHA implementation. Everyone who wishes to implement such a mechanism can use visual, audio or commonsense CAPTCHA without being concerned if the mechanism can be integrated to his VoIP application / architecture. Moreover, it can support multiple CAPTCHAs at a single implementation according to the capabilities of the devices and the preferences of the caller. So, if a caller holds a device without a screen, the CAPTCHA challenge should be audio. On the other hand, if the caller is hearing impaired or is no-native speaker then this caller must be

submitted to a visual CAPTCHA. This kind of conclusions can be exported by reading and processing the SIP messages' headers.

Furthermore, the challenge/response technique uses messages which are included in the SIP RFC 3261 and therefore they are really easy to be integrated in a SIP infrastructure. All the messages shown in figures 9 and 10 are part of the SIP handshake, except of the 419 message which is mentioned in RFC as a terminal message. The other messages, such as the one that sends the puzzle, are implemented and handled by the application, which implements the CAPTCHA.

The disadvantage of this mechanism is that the domain is responsible to produce and judge the CAPTCHA test. Therefore, the domain (proxy server) is vulnerable to Denial of Service Attacks. If a spammer, who "owes" a lot zombie PCs, starts calling a large number of subscribers of a specific SIP Provider, then the provider should create a large number of tests and consume a lot of his resources.

The other disadvantage of this technique is that the caller has to support this technique in his client. Otherwise he will get the 419 message all the time if he tries to call.

If the challenge/response (C/R) test is successful, the request will be forwarded to the callee. At this stage, the callee will answer the call or check the instant message, however, it will have also the opportunity to classify this call (respectively. Instant message) as spit or legitimate. The user feedback mechanism depicted in will allow the callee to insert the URI or the IP address used by the caller either in the black list, so further calls or messages from these user will be blacked, or in the white list, so all further calls or messages from this user will be accepted without any anti-spit detection.

3.2.2 Use case 2 : Audio analysis based solution

The audio analysis framework is another solution for dealing with spit activities. This solution follows the same spirit as the first use case, however, the authentication module and the challenge/response module are replaced respectively with the ones: proxy check module and audio analyzer module. In fact, on the one side, both the authentication module and the proxy check module are used to enforce the authentication process and prevent the use of fake identies, however, they work differently (see section 2.3.1 for more details). On the other side, the challenge/response module and the audio analyser module have the same scope, which is the detection of spit activities handled by automats. However, the detection techniques used in both are different. The authentication module filters the SIP signalling, however the audio analyzer monitors the RTP session. The audio analysis based solution is depicted in Figure 7.

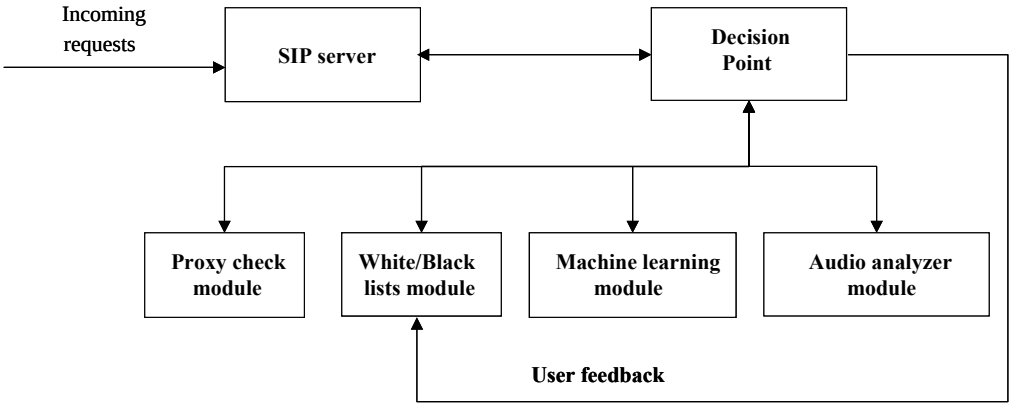

Figure 11:Audio analysis based solution

Whenever a call is intercepted by the SIP proxy, the latter forwards the corresponding request to the Decision Point. The latter, in turn, forwards it further to the first detection module in this solution, namely, the proxy check module. The latter will be used to check whether the SIP request sender is using its outgoing proxy or not. One mode of operation in SIP networks is to use the proxy of the domain not only for incoming calls, but also for all outgoing calls. If users want to make an outgoing call, they don't send their INVITE directly to the proxy of the target domain. Instead, the send it to the proxy of their own domain which than authenticates the user through the digest mechanism. This mode is so common that it may as well be called the standard mode of operation. The SIP specification calls it the SIP trapezoid.

This can be exploited for detection of SPIT. If an INVITE is received from a host that is not known to be a proxy of the domain of the caller, it is more likely to be spit and should be tested further. The Via header provides a list of all hosts that the request has traversed so far. Since it is used to route a response back, the addresses in the Via cannot be tampered with or the response will never arrive. In order to successfully establish a call, this response has to arrive, though.

A simple version of the test for an outgoing proxy is to merely count the number of Via values present in the SIP message and whether among the listed proxies in the Via headers, some are open proxies (see section 3.1.1.4). More details about how this module works will be provided in the deliverable D3.1.

If the check proxy does not report any suspicious behavior, the SIP request will be tested further through the white/black lists module and the machine learning module. The way this is achieved was already discussed in section 3.2.1. If these tests are successful, the request is directed to the last test, namely the audio analyzer module.

If the audio analysis test is successful, the request will be forwarded to the callee. At this stage, the callee will answer the call or check the instant message, however, it will have also the opportunity to classify this call (respectively, message) as spit or legitimate. The user feedback mechanism depicted in Figure 11 will allow the callee to insert the URI or the IP address used by the caller either in the black list, so further calls or messages from these user

will be blacked, or in the white list, so all further calls or messages from this user will be accepted without any anti-spit detection.

The audio analyzer module detects one of the characteristics of spit messages: the mass criterion. Each call from an unknown caller results in a checksum in a database. If a checksum is seen more than once the call is very likely a spit message. The efficiency of this technique increases with more calls that are checked. More calls results in more checksums, which contains statistically more Spit messages.

Although the basic functionality of this module was already described in section 3.1.1.7, it is important to mention that the audio analyzer module processes the SIP negotiation differently from the other modules that are already described. All other modules make their decision based on the SIP protocol, however, the Audio Analyzer bases its decision on the content of the audio stream. For that reason the call needs to be accepted, which results in a RTP session. All modules whose decision rely on the SIP protocol need to be addressed before the Audio Analyzer. It is also possible to address more modules after the audio analyzer module but they need to handle the already opened RTP session then. The Decision Point must also handle the rejection of a call differently than in all other modules. A rejection must close the RTP session as well.

The Audio Analyzer records the beginning of the call, calculates the checksum, checks the checksum against the database and returns the result. The details will be explained in the following subsections.

3.2.2.1 Audio processing and checksum calculation

The heart of the audio analyzer module is the calculation of the checksum itself. But also very important is how the algorithm is fed.

The algorithm must be able to identify the same or even slightly different audio streams as the same. This is not that difficult when the input of the audio streams is always the same. But in the VoIP environment the input of two calls where the caller streams always the same message could result in different audio data's. Due to different compressions, loss of packets, background noise or other things the audio data's could be slightly or heavily different. The algorithm must take care of such possibilities. Also important is that comparing different checksums against each other or finding a checksum in a list must be very fast. The checksum algorithm must not produce false positives in a way that two different input data results in the same checksum.

When such an algorithm is found, the spiter can try to confuse this Anti Spit technique. We report here some concrete examples,

- Changing the audio playback starting time
- Changing the audio playback speed
- Pitching the audio data
- Adding background noise or even sound

An audio preprocessor needs to be implemented which tries to normalize the audio data and feeds the checksum algorithm with it.

Figure 12: Checksums computing

3.2.2.2 Database communication

Due to the fact that the database is a standalone server application outside the Voice Analyzer, both need a way to communicate. The communication protocol is a binary protocol based on the TCP protocol. To avoid protocol version conflicts between the Analyzer and the Database, the protocol will use Key Value Pairs instead of classical structures.

Today mostly every protocol uses fixed structures to transfer data's. That implies that both communication partners have to implement all known versions of the protocol. Otherwise if each side has implemented different versions of the protocol they do not understand each other. To fix this, often a version entry will be added to the protocol but even then one of the communication partners has to implement all versions of the protocol. Normally the server has to support the newest version of the protocol and all older versions. That result in a more complicated source code on the server side and every server has to be upgraded to the newest protocol version before any client can use a new version. The solution is to use a protocol, which is up and downward compatible. The Analyzer and the Database use a Key Value Pair based protocol to ensure this compatibility.

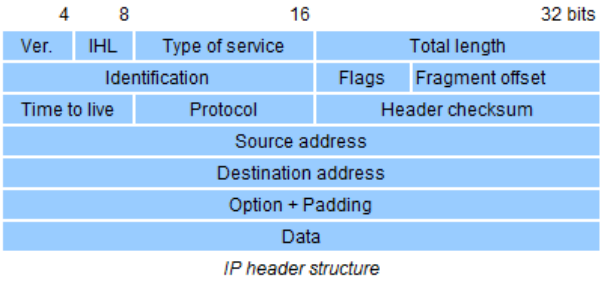

Figure 13:Example of a fixed protocol structure

A Key Value Pair based protocol is not fixed at all. The way how to transfer these Key Value Pairs in requests and replies is fixed but not the values keys, values nor the type of the values. One side sends all the keys/values, which are defined in its version, the other side just look on these keys which are known in his version. This can result in three different situations:

1. The receiver interprets every key (both versions are equal)
2. The receiver skips unknown keys (may be because of added keys in a newer version)
3. The receiver misses some keys (may be because the sender does not know these yet)

The implementation of both sides must handle the third case sufficient. This is very easy when default values are assumed if some key is missing.

Each request or reply starts with an identifier and the total size of the packet. A Key Value Pair will start with the Key followed by the type of the value, the total size of the value and the value itself.

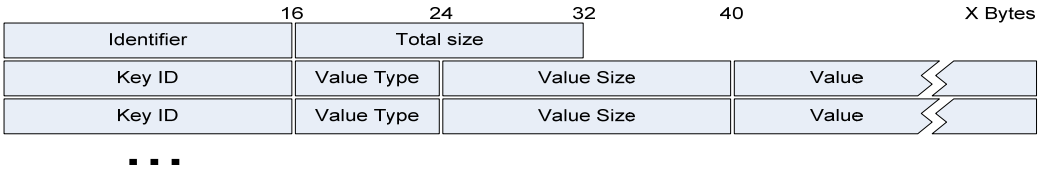

Figure 14: The use of key value pairs

3.2.2.3 Checksum Database

The Checksum Database is the mind of this module. It receives the requests with the checksum from the Audio Analyzer, looks for the checksum in its memory, increments the counter and sends the reply back to the Analyzer.
All checksums will be hold in RAM in a list or hash tree where the search can be performed very quickly. Each time, a checksum is found or created, the counter of that checksum will be incremented and the modification date is changed. Checksums that are older than a certain amount of time and the corresponding counter is one, can be removed from the list because the mass criteria was not reached. Checksums with counter greater than one could also be removed if they were not updated for a much greater period of time (the recommendation is a time period greater than one month). Deleting old entries will depend on the traffic volume. The more traffic is checked, the earlier checksums can be removed. The suggested procedure is to reserve a certain amount of RAM for the list of checksums; when that amount is reached the delete procedure should start and delete 10-15% of old checksums.

For stability reasons, checksums with counter greater than one should be saved to a SQL database regularly. The Checksum Database should initialize at the startup the checksum list from that SQL database.

When more than one Checksum Database is needed they should synchronize themselves through the SQL database. New or updated checksums in the SQL database need to be updated in RAM and updated checksums in RAM with counter greater one need to be updated in the SQL database.

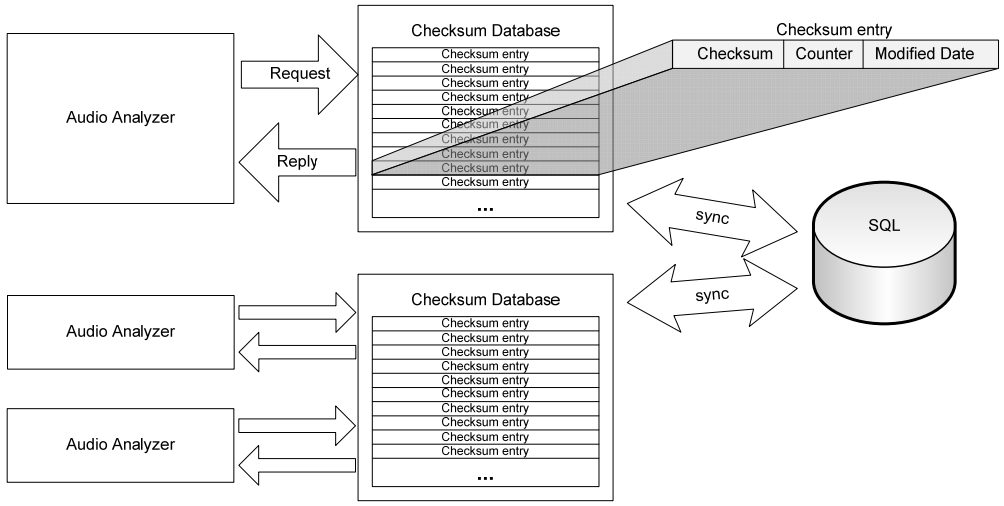

Figure 15: Checksums storage procedure

3.2.3 Use case 3 : Reputation based solution

Reputation is in general the opinion of the public towards a person, a group of people, or an organisation. This concept is an important factor in many fields, in particular, networking and online communities. The solution we are going to describe in this section uses the chain of trust that may exist between the SIP request receiver and the SIP request sender as described in section 3.1.1.6.

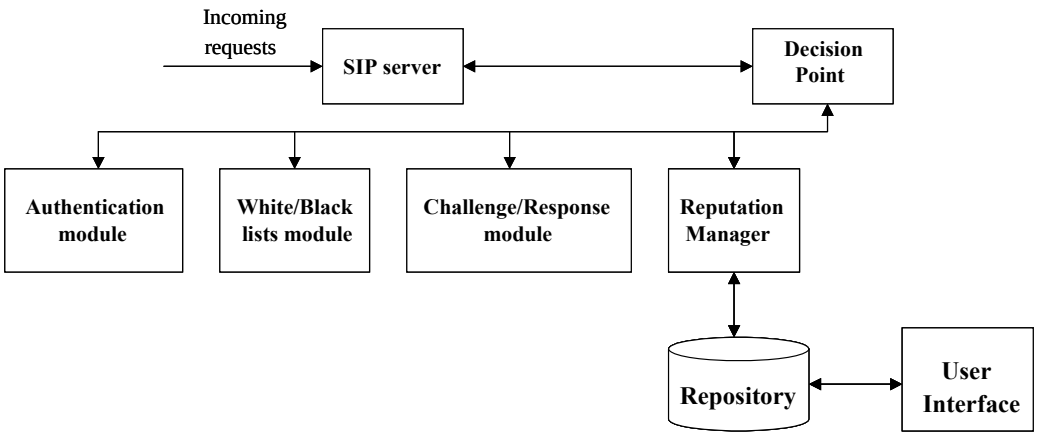

Figure 16:Reputation based solution architecture

The architecture of the reputation-based solution is depicted in Figure 16. It includes the main following components,

3.2.3.1 Reputation Manager

Already described in section 3.1.1.6.

3.2.3.2 Repository

In this technique, every SIP user has the possibility to set his own contactlist. The latter is simply a set of SIP users' identifiers such as SIP URLs with some corresponding values ranging between 0 and 1 and which reflect the amount of trust that the list owner has in these persons. The contact lists repository can be accessed through a simple user friendly interface.

3.2.3.3 Solution description

The first detection module in the reputation based solution if the authentication module. The latter will assure that the caller is not using a spoofed IP address or a fake domain name as described in the use case 1. If the operation triggered by this module (see section 3.1.1.1) fails, the call will be dropped. If this operation succeeds or if one of the domains (inbound or outbound) is not implementing this technique, the call will be treated as a normal SIP transaction, so it will be submitted to the next anti-spit test, which is represented by the white/black lists module. The way this module works was already discussed in section 3.2.1.

The white/black lists can be set up simply by using an interface allowing the users (or the administrators) to set their preferences. However, it also possible that these lists can be established in an automatic fashion. In fact, the Reputation Manager can look up regularly in the SIP users contact lists for users whose reputations values are below a lower threshold *h1* or greater than an upper threshold *h2*. If a given SIP user is assigned a reputation value, either less that *h1* or greater than *h2*, by a "reasonable" number of contact lists owners, this user will be declared as a spammer and will be added to the users black list or respectively as a legitimate SIP user and he will be added to the users white list. The thresholds *h1* and *h2* can be set for instance to the values 0.2 and 0.8 respectively. To achieve this task another parameter is needed which will reflect the number of contact lists owners that when exceeded, the scored SIP user is added either to the users black list or users white list.

If the SIP request sender does not belong to the white and black lists, this request will be submitted to some other tests. As most of spam is generated by machines, our security framework has to distinguish between machines and human beings. For this reason, the challenge/response technique is used here. When this technique is used, a challenge is sent to the sender of the message and the latter is processed if and only if the sender answers to the challenge correctly. This technique is discussed in details in section 3.1.1.5. If the SIP request fails to pass the test under consideration, this means that the request sender is likely to be an automat. In this case, the request will be rejected in the same way as described earlier. If the SIP request successfully passes the challenge/response test, this means that the request sender is probably a human being and not an automat. As this stage, the Decision Point will be informed and the request will be challenged against the reputation mechanism process, which is described in section 3.1.1.6. If a path between the request sender and the destination is found, the corresponding reputation value is compared to a predefined threshold. If this value is greater than the threshold, the request will be processed, otherwise, it will be rejected. In case, the user receiving the request classifies it as spam, the user has the possibility to insert the request sender information in the black list. This will block any further request coming from this user in the future. The request receiver also has the possibility to insert the request sender in the white list in the same way as described for the black list. The third possibility is

to insert the request sender into the request receiver contact list and assign it a reputation value, which will be used for any further request coming form this sender.

4 Conclusion

In this deliverable, we have provided a high level architecture for dealing with the spit problem. After assessing the spit risk and evaluating the current anti-spit solutions, the major issues behind the emergence and the persistence of spit are extracted. From these issues, some requirements for anti-spit solutions were defined and a generic architecture was suggested. This architecture is also modular and aims at combining different classifiers in order to achieve the best discrimination between spam and legitimate SIP requests. The modular architecture we have adopted would allow the proposed anti-spit solutions to be updated, upgraded and adjusted to the needs of the providers without any difficulty. In this architecture, we have identified two main layers. A detection layer where the spit detection intelligence is distributed over different modules, and a decision layer where the results received from the detection layer are gathered and accordingly a decision is made.

Once the mentioned modules were introduced, some case studies of the generic architecture were provided. The reason was just to show that possible efficient solutions did exist and a selection between the intelligent modules was possible according to the providers' requirements and preferences.

5 References

[RFC3261] J. Rosenberg et Al, "SIP: Session Initiation Protocol", RFC 3261, June 2006

[SCH06] D. Schwartz et Al, "SPAM for Internet Telephony (SPIT) Prevention Using the Security Assertion Markup Language (SAML), link: draft-schwartz-sipping-spit-saml-01.txt

[CRO05] N. Croft, M. Olivier, "A Model for Spam Prevention in Voice over IP Networks using Anonymous Verifying Authorities," in Proc. of the 5th Annual Information Security South Africa Conference (ISSA 2005), South Africa, July 2005.

[MAT06] B. Mathieu, et al., "SPIT Mitigation by a Network-Level Anti-Spit Entity", in Proc. of the 3rd Annual VoIP Security Workshop, June 2006, Germany.

[REB06] Y. Rebahi, D. Sisalem, T. Magedanz,, "SIP Spam Detection", in Proc. of the International Conference on Digital Telecommunications, pp. 29-31, August 2006, France.

[SRI04] K. Srivastava, H. Schulzrinne, "Preventing Spam For SIP-based Instant Messages and Sessions", Technical Report, University of Columbia, 2004.

[DON06] S. Dongwook, A. Jinyoung, S. Choon, "Progressive Multi Gray-Leveling: A Voice Spam Protection Algorithm", IEEE Network, 20(5), pp. 18-24, Sept./Oct. 2006.

[BAU06] R. Baumann, S. Cavin, S. Schmid, "Voice Over IP - Security and SPIT", Swiss Army, FU Br 41, KryptDet Report, Univ. of Berne, September 2006.

[RFC4474] J. Peterson, and C. Jennings, "Enhancements for Authenticated Identity Management in the Session Initiation Protocol", RFC4474, August 2006.

[TSC06] H. Tschofenig, R Falk, J. Peterson, et al., "Using SAML to Protect the Session Initiation Protocol", IEEE Network, 20 (5), pp. 14-17 September/October 2006.

[MAD06] Madhosingh, "The Design of a Differentiated SIP to Control VoIP Spam", Technical Report, Computer Science Department, Florida State University, 2006.

[NICO] S. Niccolini, "SPIT prevention: State of the art and research challenges", Network Laboratories, NEC Europe, Germany.

[DAN05] R. Dantu, P. Kolan, "Detecting Spam in VoIP Networks", in Proc. of Steps to Reducing Unwanted Traffic on the Internet Workshop (SRUTI '05), July 2005, USA.

[MAR07] Marias J., Dritsas S., Theoharidou M., Mallios J., Gritzalis D., "SIP vulnerabilities and antispit mechanisms assessment", in Proc. of the 16th IEEE International Conference on Computer Communications and Networks (ICCCN '07), IEEE Press, Hawaii, August 2007 (to appear).

[SIP3261] J. Rosenberg, et al., Session Initiation Protocol, RFC 3261, June 2002.

[RTP1889] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, RTP: A Transport Protocol for Real-Time Applications, RFC 1889, IETF. January 1996.

[ROS06] J. Rosenberg, C. Jennings, The Session Initiation Protocol (SIP) and Spam, draft-ietf-sipping-spam-03, October 2006.

[MEH06] V. Mehta, C. Bartzis, H. Zhu, E. Clarke, J. Wing, "Ranking Attack Graphs", in Proc. of Recent Advances in Intrusion Detection, pp. 127-144, LNCS Springer-Verlag, Germany, September 2006.

[DRI07] Dritsas S., Mallios J., Theoharidou M., Marias G., Gritzalis D., "Threat analysis of the Session Initiation Protocol, regarding spam", in *Proc. of the 3rd IEEE International Workshop on Information Assurance* (in conjunction with the 26th IEEE International Performance Computing and Communications Conference (IPCCC-2007), pp. 426-433, IEEE Press, New Orleans, April 2007.

[RFC3515] R. Sparks, "The Session Initiation Protocol (SIP) Refer Method", RFC 3515, April 2003