A framework for detecting anomalies in VoIP networks

Abstract— **This paper introduces a novel system (architecture and techniques) that aims to secure overlay networks by detecting anomalies that can be a priori known or new for Voice over IP networks. It is particularly designed for the signaling protocol SIP. The proposed system mainly consists of two parts. The first one determines the different features that are extracted from the specification of the SIP protocol. In fact, these features should highly characterize the behavior of the signaling traffic so that the evidence of the intrusion is not lost when only these attributes are considered for the attack detection goal. After the attributes extraction step, a detection algorithm is used to classify new SIP profiles in their appropriate class (either as normal, or as an anomaly). Another feature of this system is its adaptability since a feedback from the detected attacks is possible.**

I. Introduction

Voice over IP (VoIP) networks play increasingly a vital role in current IP networks infrastructure for the modern society. SIP (Session Initiation Protocol) is the widely (signaling) application layer protocol that is used to create, modify and terminate a session with one or more participants and may be used for voice, game, instant messaging and visio-conference applications. SIP has been chosen by many groups such as 3GPP [1] for establishing multimedia sessions and has seen many commercial and free software implementations. Although this protocol has seen many developments and a big interest from the telecommunication community, it suffers from many security flaws and faces new attacks not a priori known. The attacks described in the recent academia and research papers against this new emerging protocol are not exhaustive and are generally inspired from the known attacks that targeted the TCP/IP networks during the last three decades.

Access control techniques using SBCs (Session Border Controllers) and cryptography mechanisms are used in this new environment to filter and counter some anomalies. However, these solutions are not sufficient since there are many attacks that easily bypass these mechanisms. As in the IP infrastructure, intrusion detection is considered as a second barrier to secure information systems.

Intrusion detection systems (IDSs) are widely used in commercial and governmental information systems. The different IDSs focused on either pattern matching techniques or on some entity behavior learning. Generally, pattern matching techniques try to recognize patterns in the packet header or in the payload. Methods based on the entity behavior learning use some classification techniques that consider statistical measures. In their initial form, these measures consisted of monitoring the traffic to a protected resource or the traffic from a particular IP address. However, little intrusion detection research work has been done for VoIP systems. The research work done in this field, unfortunately, uses the same methods implemented for TCP/IP traffic.

Since the different intrusion detection techniques that are implemented until now are not appropriate to detect the different attacks targeting the current VoIP systems, novel techniques should be introduced. The main idea behind our proposal consists in considering the process of intrusion detection as a three-stage process. The first stage consists in collecting the VoIP traffic that is either safe, which is free of attacks, that we call here normal and attack traffic that contains traces of attacks evidence. The second stage consists in extracting attributes; those features that keep the most information characterizing the traces without attack and normal traffic evidence loss. The last one is the classification process that is based on a model able to distinguish between normality and abnormality. This model is built over a set of traces where the corresponding traffic is either labeled as normal or as an attack within the different a priori known VoIP attacks.

The rest of the paper is organized as the following. Section II presents the different research works done recently to detect intrusions in VoIP networks. Section III discusses the principal components of our framework. Section IV presents some intrusions that we developed to attack a real VoIP infrastructure. Section V depicts the environment of the different experiments we conducted and the different results obtained. Finally, Section VI presents future work and concludes the paper.

II. Related work

Intrusion detection research for VoIP networks is currently at its infancy stage. In our knowledge, the research works done in this direction use the same basic methods implemented during the last three decades for detecting intrusions in the TCP/IP traffic.

Some researchers use the same directions as those of the Snort IDS [12] which is based on a pattern matching technique that looks over packets' streams for recognizing patterns in the packet header and/or payload. Others use some classification techniques that consider statistical measures. In their initial form, these measures consisted in monitoring the traffic to a protected resource or the traffic from a particular IP address.

For the first case, we cite the "*Scidive*" and "*Spacedive*" presented in [18]. These two basic systems are based on a simplistic correlation engine between the events of the signaling and the media stream protocol to detect a few types of attacks. They are also based on the Snort detection engine where only a simple extension is done for stateful and cross-protocol detections.

For the second case, a team in LORIA [9] uses the same method as that of Skinner and Valdes presented in [17] which is a Bayesian model called TCP EBayes. While TCP Ebayes uses only the TCP protocol to detect anomalies, the authors in [9] use the SIP protocol to detect the same basic anomalies as those targeting TCP such as syn-flooding and port scanning. Therefore, instead of using the number of open TCP connections, the number of unique IP addresses and the number of unique ports as in TCP EBayes to detect port scanning and IP sweeping, the number of open RTP ports, the maximum number of waiting dialogs, etc. are used. There are many problems related to this technique. As an example, only bursts of traffic are considered as evidence of an anomaly. As a result, only the flooding attacks may be detected. In addition to this, the system was not experimented for the VoIP network case due to the lack of a real testbed. The original goal of the TCP EBayes is to detect abnormality; that is the detection is binary. This is not an appropriate method in particular for an overlay networks application where the administrator or the operator should be informed about the type of the attack for the next stage that consists in launching an appropriate counter measure.

Recently, the state machines are used to detect some intrusions in VoIP network [15]. The proposed approach utilizes not only the state machines of network protocols but also the interaction among them. However, the different attacks tested by this mechanism are simplistic since there is no an in-depth study of the SIP protocol and almost all the defined attacks are launched by a third party. In an operational network, these attacks are hard to perform because of the different security mechanisms that are made in place by the telco operator such as those defined by the 3GPP [1]. However, these attacks are only possible in a LAN (Local Area Network) without any security mechanisms or a simulated network as experimented in [15].

III. Systematic framework

Since the different IDS techniques that are starting to come up with the emerging VoIP protocols are in their infancy stage on one hand or use the same vulnerable techniques as those implemented during the last decades on the classical IP networks on the other hand, we have to introduce novel techniques to detect the real intrusions that focus mainly on the new emerging VoIP protocols. In the following, we present a novel architecture that is able to detect anomalies and to correctly classify normal signaling traffic generated by the current VoIP networks.

There is a variety of goals for this mechanism. First, it detects the whole a priori known attacks by an automatic learning. Second, it easily discriminates the different attacks and the safe VoIP traffic. Third, it recognizes new anomalies; those that are not learnt during the first step. These new anomalies may be due to the new vulnerabilities discovered and exploited by potential attackers. In addition, this system is a complete one since it does not only detect attacks but also focuses on the relevant VoIP features that should be considered for the detection goal. Another dimension of this mechanism is that it does not only use a stateful detection technique but also looks at different protocols used for establishing and maintaining the VoIP communications. Moreover, it generates statistical measures, corresponding to the different features, between the current packet (resp. transactions or dialogs) and the last packets (resp. transactions or dialogs) for the goal of VoIP intrusion detection. Finally, It is an extensible mechanism because it is able to learn the different classes of traffic (normal or attack) and adaptively consider new attacks and new normal forms by simple updates. It is also insensitive to IP spoofing and handles client mobility.

We mention that this mechanism is used as a first step before launching counter measures. Once the attack is detected, it sends to the corresponding reaction mechanism (as a future work) the different features that characterize the traffic that has caused the intrusion for appropriate counter measures.

We notice that this mechanism is implemented either in a device or as a logical module placed in front of a user agent; be it a client or a server, or in front of a VoIP server (a proxy server or a registrar). The only condition for this mechanism is the ability to catch all the inbound and outbound traffic of the monitored VoIP equipment. It may also be implemented behind or in front of a firewall with or without a NAT to which it is transparent.

A. Framework architecture

The different components of the proposed system are depicted in Figure 1. The first part consists in defining a profile that corresponds to a set of attributes that summarizes a VoIP flow and catches the evidence of normality and anomaly.

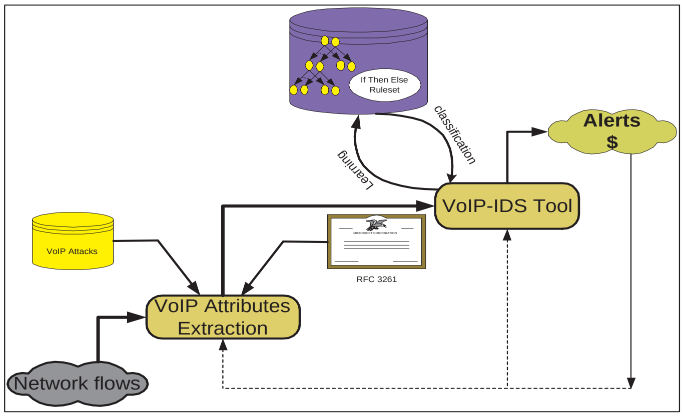

Fig. 1
VoIP intrusion detection architecture.

We define three profiles for the goal of characterizing VoIP traffic and catching VoIP intrusion evidence. The first corresponds to a set of attributes extracted from packets and the different measures that correlate the current packets with the last ones as explained below. The second is a transaction based one. The third is a dialog based one. For this last profile we do not only consider signaling and description protocols but also the RTP and other protocols that are used for media transfer. This latter is complementary to the cross protocol used by "*Scydive*" [18] because we do not write manually simple signatures and other attributes considering media flows are taken into account.

B. Different components

The first step of the system consists in extracting the different attributes that characterize the different attacks and the normal network flows. These attributes are extracted by using a set of known VoIP attacks based on SIP according to its specification as defined by RFC 3261 [13]. As mentioned above, we determine three different profiles that are used to characterize the SIP signaling flows. The first is the *packet-based* where each flow corresponds to a set of attributes extracted from packets and the different measures that correlate the current packet being analyzed with the previous packets. The second profile is based on *transactions*. A transaction, as defined in RFC 3261 [13], consists of a request that invokes a particular method, or function, on a server and at least one response. We note that SIP is based on an HTTP-like request/response transaction model. The third profile is based on a dialog. A dialog is a peer-to-peer SIP relationship between two user agents that persists for some time. The dialog facilitates sequencing of messages and proper routing of requests between the user agents. The INVITE method is the only way defined in RFC 3261 to establish a dialog. The *dialog-based* profile corresponds to a session where not only signaling and description protocols are considered, but also RTP [14] and other protocols that are used for media transfer. The third profile is complementary to the cross protocol used by *"Scidive"* [18].

Due to space limitation, we only present in the following the different experiments and results when considering the packet-based profile. The method does not differ between the three determined profiles. However, only the set of attributes is different from one determined profile to another. Notice that a combination of these profiles by merging the three profiles into a single one containing the union of all attributes of these three profiles may lead to another technique. Combining the different alerts generated by each profile may also lead to a new technique.

For the attributes extraction step, we define two different kinds of attributes. The first set of attributes comprises attributes as defined in RFC 3261 [13] related to SIP. Furthermore, these attributes are extracted based on the known attack types. The first set of attributes is extracted manually by a security expert directly from RFC 3261, thus the attributes of the first set are called intrinsic attributes.

The second set of attributes is automatically extracted from the first one. This latter corresponds to different statistical measures between the current network flow and the past flows according to a time window having a length of N or according to a window of M SIP flows, where N is a positive value and M is a positive integer. The second set is automatically constructed from the first set by considering intrinsic statistical measures between the current flow attributes and those of last flows contained in a window of N seconds or only by considering the last M flows. The values of N and M are fixed by experience. For instance, a period of 2 seconds is used for the time window and 200 flows preceding the current one are used for the other window. The intrinsic attributes can be defined to belong to a first class, the attributes related to the time window are defined to belong to a second class and the attributes related to a window of M flows are defined to belong to a third class. Thus, the attributes of the first class belong to the first set, whereas the attributes of the second and third class belong to the second set. The attributes of the second set can equally be called expert knowledge attributes, since a security expert determines the attributes that belong to this set.

Attribute extraction is an essential task before applying the detection process. As a prior work, Lee et al. [7] used directly the Bro [10] tool as a packet filtering and connection reassembling engine to extract the different attributes. The KDD99 intrusion detection database [6] is built upon this basic extraction whose goal is the detection of basic attacks over TCP/IP. We note that more than one hundred research papers used this database to assess their proposed intrusion detection technique and all the detection methods failed to detect some attack categories. This failure is not due to the detection method but to the infomation loss while transforming raw traffic into attributes connection records (for more details on the KDD information loss, see for instance [3], [2]). As a matter of fact, we have taken a lesson from this experience and we deeply analyzed the SIP protocol using the specification defined by RFC 3261 [13] and the known attacks related to VoIP networks, discussed in Section IV, for extracting meaningful attributes for the goal of VoIP intrusion detection.

The second step of the proposed mechanism is the detection process that uses as input the profile extracted from the network flows as described above. Once the profiles are determined, the detection step could be thought of as a classification problem: we wish to classify each profile into one of a finite set of possible categories; normal, one possible attack type, or a new observation probably corresponding to a new attack. Given a set of profile records, where one of the features corresponds to the class label of the profile (i.e. normal, attack or new), classification and induction algorithms can construct a model that is able to summerize each category by using the most significative attributes to each category. Notice that it is also possible to use unsupervised classification techniques to classify the profiles. However, in the unsupervised techniques the classification is binary (normal or abnormal) according to some assumptions a priori taken such as the normal profiles are almost gathered in a dense region if projected to a two axes' space for example, and outliers are considered as attacks. We do not recommend to use this technique since a priori known attacks should be learned in order to not confuse them with normal flows.

The proposed mechanism is experienced using supervised classification techniques. In fact, a set of knwon attacks is played against a SIP user agent; may it be a server or a client. The corresponding flows generated from each attack are labeled with their appropriate attack type. The normal traffic is collected from a real world infrastructure of a telecommunication operator.

When training the classification model with a learning database containing a variety of attack and normal flows, a feedback from the detected attacks is used to improve the successful detection rate. As a matter of fact, in the case where some attacks are not detected (false negatives) or some normal traffic is classified as an attack (false positives) then an expert is in charge in checking whether other attributes should be considered, or this misclassification is due to the second stage (i.e. the detection model). The reason for taking other attributes into consideration consists in lessening the information and intrusion evidence loss when transfomring the raw network traffic into a set of attributes. However, if the misclassification is due to the classification process then the classification technique should be tuned to increase the successful detection rate of the different tested flows belonging to the learning database, for instance.

C. Detection models

In the proposed mechanism, we call a detection model the method that learns automatically the different samples present in the learning database. As a result of the learning step a classification model is built with which new unlabeled instances are classified in their appropriate category (attack type or normal). If the corresponding class is an attack then an alert is generated, otherwise the flow is considered as normal. Since we use a learning database in which all flows are labeled in their appropriate class, we may use different supervised classification techniques for the task of the building process. There are many candidate techniques available in the data mining literature. In the following, we focus on decision trees induction algorithm as the technique for learning labeled flows and classifying new ones for the detection goal. However, any other supervised or unsupervised one may be used for this goal. Decision trees classifiers are based on the “divide and conquer” strategy to construct an appropriate tree from a given learning set S containing a finite and not empty set of labeled instances.

The decision tree is constructed during the learning phase, it is then used to predict the classes of new instances. Most current decision trees algorithms use a “top down strategy”; $i.e.$ from the root to the leaves. Two main processes are necessary to use the decision tree:

- **Building process** It consists in building the tree by using the labeled training data set. An attribute is selected for each node based on how it is more informative than others. Leaves are also assigned to their corresponding class during this process.
  To measure how informative a node is, Shanon entropy is used to construct the decision trees. The selection of the best attribute node is based on the gain ratio $GainRatio(S, A)$ where S is a set of records and A a non categorical attribute. This gain defines the expected reduction in entropy due to sorting on attribute A. It is calculated as the following [8]:

$$Gain(S,A) = E(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} E(S_v) \quad (1)$$

  where $E(S)$ corresponds to entropy of S.
  In general, if we are given a probability distribution $P = (p_1, p_2, .., p_n)$ then the information conveyed by this distribution, which is called the Entropy of P is :

$$Entropy(P) = -\sum_{i=1}^{n} p_i log_2 p_i \quad (2)$$

  If we consider only $Gain(S, A)$ then an attribute with many values will be automatically selected. One solution is to use $GainRatio$ instead [11]:

$$GainRatio(S,A) = \frac{Gain(S,A)}{SplitInformation(S,A)} \quad (3)$$

  where

$$SplitInformation(S,A) = -\sum_{i=1}^{c} \frac{|S_i|}{|S|} log_2 \frac{|S_i|}{|S|} \quad (4)$$

  where S_i is a subset of S for which A has a value v_i.
  This partitioning strategy is used to build the tree, having as a main goal to divide the considered training example by selecting recursively the best non categorical attribute. In the case of a discrete valued attribute, this strategy tests all possible values of the attribute under consideration. However, in the case of continuous-valued attributes a transformation technique is introduced [11]. It consists in defining new discrete-valued attributes that partition the continuous attribute into a discrete set of intervals. The algorithm dynamically creates a new boolean attribute A_t that is true if $A < t$ and false otherwise. The selection of the threshold value t is based on the information gain (see Equation (1)). A threshold t is selected if it produces the greatest information gain. The different items according to the continuous attribute A are sorted, then a set of candidate thresholds midway between the corresponding values of A is generated. Fayyad [4] showed that the value of t that maximizes information gain lies always at such a boundary. These candidate thresholds are evaluated by computing the information gain associated with each of them. The dynamically created boolean attributes can then compete with the other discrete valued candidate attributes that are available for growing the tree. In the following, we use this partitioning technique for evaluating the attributes with continuous values.
- **Classification process**: A decision tree is important not because it summarizes what we know, i.e. the training set, but because we hope it will classify correctly new cases. Thus, when building classification models, one should have both training data to build the model and test data to verify how well it actually works. New instances are classified by traversing the tree from the root to the leaves based on their attribute values and the node values until one leaf is reached that corresponds to the class of the new instance.

Besides the construction and classification steps, many decision trees algorithms use another optional step. This step consists in removing some edges that are considered useless for improving the performance of the tree in the classification step. Pruning trees simplifies the tree since many useless edges are removed making complex trees more comprehensive for interpretation. In addition, a tree that is already built is pruned only when it gives better classification results than before pruning [8].

In Section V-C, we give some examples of the decision tree obtained from the different experiments we conducted over the different attacks presented in Section IV.

We note that the building process is done off-line while the detection process may be performed either on-line or off-line depending on the security policy of the information system.

IV. The considered attacks

SIP is widely used in VoIP systems and there are numerous attacks that can be performed against the SIP signaling protocol. The attacks are ranging from syntactical attacks; those that do not follow the SIP grammar provided by RFC 3261, to different denial of service (DoS) attacks in the overlay networks. Other attacks are the same as those that exploit known flaws such as buffer-overflows against servers. Only the attacks that affect directly the signaling protocol are investigated since the syntactical attacks and different flaws that are due to the programming errors have been widely investigated and current IDSs detect a variety of these attacks. In the following, different attack types corresponding to SIP attack scenarios are discussed. These attacks can be divided into three categories namely; information gathering, service theft and DoS.

In the following, we list representative attacks we investigated that we gather into the three categories. We give for each category its significance and some flow examples of the corresponding SIP attack Scenario. Notice that other attacks such as covert channels, SIP directory scanning, QoS degrading are investigated but are not listed below due to space limitation.

A. Information gathering

Generally, an attacker has to perform many actions in order to achieve her malicious goal. These actions correspond to an attack scenario composed of many elementary attacks. Information gathering is one type of these elementary attacks, where the attacker may first collect information about the target server to get its version to check whether there is any known vulnerability to exploit. The attacker may also seek for some security credential variable variations such as nonce variation where the second step of this attack scenario might be a replay attack. Password guessing and directory scanning correspond to other information gathering attack types. For instance, the directory scanning attack, which involves checking for existing valid user identities in the registrar database, may be followed by a password guessing attack since a valid username was found.

1) Nonce variation determining: According to RFC 3261 [13], SIP provides a stateless challenge based mechanism for authentication brought from HTTP authentication provided by RFC 2617 [5]. The *“Digest”* authentication is introduced in SIP for message authentication and replay protection only and without considering message integrity or confidentiality. One credential variable of this mechanism is the *“nonce”* that is used to compute the hash value of the authenticated response message using for example the MD5 hash algorithm. To check whether replay attacks are possible, the attacker may check if the nonce is changed for every authenticated message or it is renewed periodically, say for instance once every second. In this last case, replay attacks remain possible. To perform this attack, the attacker may send many requests during a short period of time, say for example one second. Figure 2 shows such an elementary attack where the attacker tries to find out the randomness of the nonce value by sending a burst of REGISTER requests to the target server, say for example 20 REGISTER requests per second, and checking the values of the nonce in the server messages corresponding to the WWW-Authenticate header field. We mention that this attack may be also performed using other request methods such as INVITE, etc. This attack is also possible against a proxy where the authentication challenge is extracted from the Proxy-Authenticate header field.

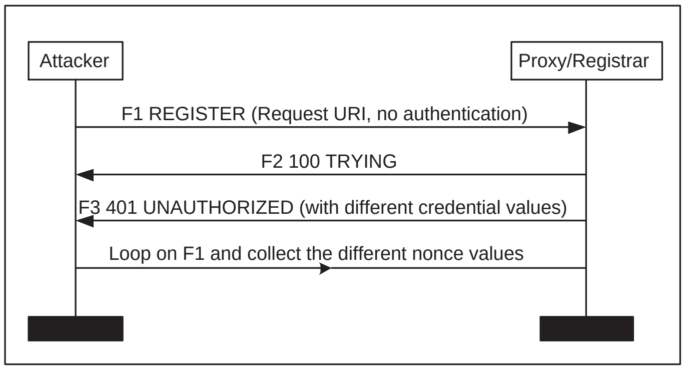
Attacker
Proxy/Registrar
F1 REGISTER (Request URI, no authentication)
F2 100 TRYING
F3 401 UNAUTHORIZED (with different credential values)
Loop on F1 and collect the different nonce values

Fig. 2
Nonce variation determining attack.

2) Directory scanning: This elementary attack consists in collecting valid identities corresponding to legitimate clients in the operator databases. It may be performed using different SIP message flows. It is considered as an information gathering attack since we only try to find valid URIs for a further malicious intention. It may be considered as the step that precedes another elementary attack such as an identity theft by using a dictionary to guess the corresponding password of the identity that was discovered during this first stage. We should mention that this attack may be omitted particularly for those identities that are in the red list. In fact, the corresponding operators may add appropriate mechanisms for such lists. However, this attack is tested against many plateforms of different operators and the experiments are successful.

Figure 3 shows a possible SIP scenario flow that may be used to perform this attack. According to the first messages exchange, a “401 Unauthorized” response is received when the identity corresponds to a known valid user whereas “403 Forbidden” is received in the other case. Therefore, an attacker may repeat this scenario and according to the response, she concludes whether the requested identity is valid or not. This attack may be also performed using the OPTION request method. In fact, according to the response, one can know whether the corresponding URI mentioned in the “To” header field is valid or corresponds to an unknown user.

B. Identity and service theft

While the above attack consists in collecting information about users and servers, this attack kind consists in stealing the identity of a legitimate user that either has mistakenly left his password unprotected for different reasons or an attacker has in-

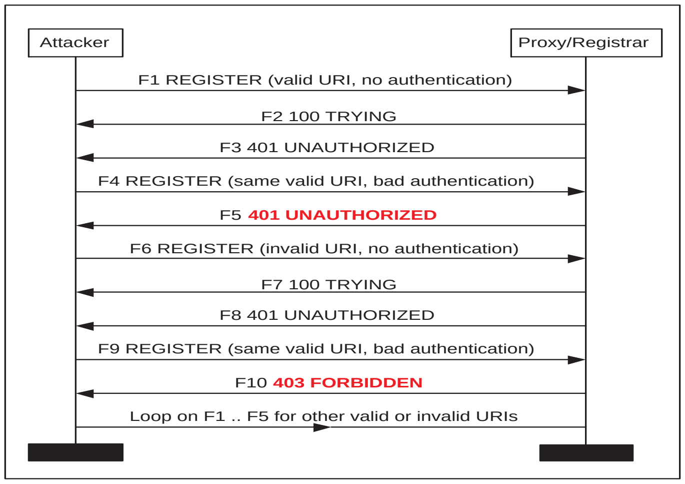
Attacker
Proxy/Registrar
F1 REGISTER (valid URI, no authentication)
F2 100 TRYING
F3 401 UNAUTHORIZED
F4 REGISTER (same valid URI, bad authentication)
F5 401 UNAUTHORIZED
F6 REGISTER (invalid URI, no authentication)
F7 100 TRYING
F8 401 UNAUTHORIZED
F9 REGISTER (same valid URI, bad authentication)
F10 403 FORBIDDEN
Loop on F1 .. F5 for other valid or invalid URIs

Fig. 3
DIRECTORY SCANNING ATTACK.

tentionally cracked his password by performing appropriate attacks such as those based on dictionary or moreover any brute force technique. Another kind of this attack type consists in using a service to which the user is not authorized or to which she is not subscribed.

1) Password guessing: One well known attack uses a dictionary to find out a user password, or a brute force technique by exploring a large number of possibilities. Therefore, an attacker may use a series of passwords for a specific identity, discovered during the last stage. She may succeed to discover the correct password of this entity in particular when the corresponding user has not chosen an appropriate password. Figure 4 shows the scenario corresponding to this attack.

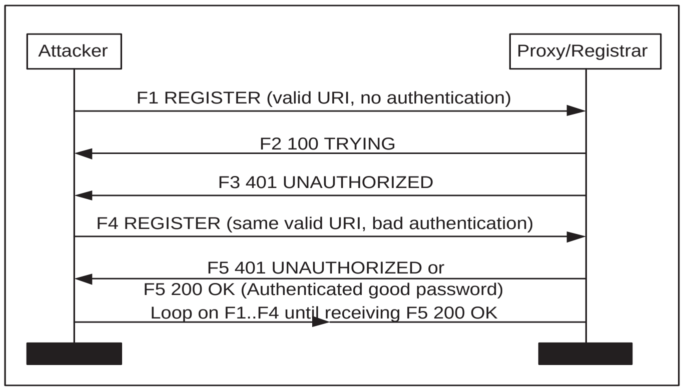
Attacker
Proxy/Registrar
F1 REGISTER (valid URI, no authentication)
F2 100 TRYING
F3 401 UNAUTHORIZED
F4 REGISTER (same valid URI, bad authentication)
F5 401 UNAUTHORIZED or
F5 200 OK (Authenticated good password)
Loop on F1..F4 until receiving F5 200 OK

Fig. 4
PASSWORD GUESSING ATTACK.

C. Denial of Service

The DoS attack is a technique that is largely used since the introduction of computers. Its goal is to make a physical resource unavailable to its legitimate users. This kind of attack can be divided into two categories. The first one is based on the flooding DoS whereas the second one involves sending a malformed packet that causes the endpoint to crash. When performing the DoS attack, an attacker can send a huge number of successive REGISTER requests against a registrar or many INVITE requests to a target client. On the other hand, an attacker may follow the dialog when sending the INVITE to a legitimate client and can stop the flow of the SIP signaling by sending a BYE request just after he receives the OK response from the target client.

1) DoS against a server: A DoS attack against a server is a flooding attack that involves sending a non restrictive number of requests against a server such as a registrar. This type of attack may be also extended to a distributed DoS (DDoS) attack where the attacker recruits many zombies over the Internet and each compromised machine sends huge numbers of such legitimate requests.

2) DoS against a legitimate client: When performing a DoS attack against a legitimate client, an attacker tries to disturb a legitimate client based on continuous INVITE requests without establishing the call since the attacker cancels the call each time the user answers the request. Figure 5 shows the corresponding attack scenario.

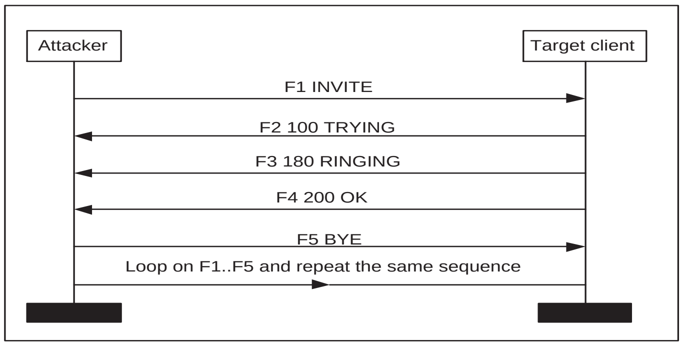

Fig. 5
DOS AGAINST A CLIENT ATTACK.

V. EXPERIMENTS

A. The environment setup

We participate in a project that aims to detect anomalies in overlay networks with an international telecommunication group as a partner. We were provided with a *tcpdump* traffic of 2 hours collected from an operational testbed. The collection was done downstream of an SBC (Session Border Controller). This collection was done in November 2006 where approximately 1640 clients used the VoIP SIP testbed during this period. The result after filtering the *tcpdump* collection and keeping only the traffic corresponding to SIP and RTP protocols consists of about 200 MBytes per each hour. We manually and meticulously analyzed all the packets corresponding to the SIP protocol and found that there are some syntactically malformed SIP packets according to the SIP grammar provided by RFC 3261 [13]. We filtered the corresponding packets since we do not consider this attacks kind as explained in Section IV. We then assumed that the filtered collection is free from signaling attacks and conducted our experiments by injecting the attacks described in Section IV into the collected set. In fact, we implemented a tool that behaves as a user agent client that launches different attacks, against a VoIP overlay network infrastructure using SIP as the signaling protocol, such as those presented in Section IV.

We used the first data set corresponding to the first hour as a learning dataset after having peppered it with attacks that are launched against the operational infrastructure. Notice that there are machines that are connected to this infrastructure playing the role of attackers. The different attacks that are launched against the infrastructure are successful. As for example, the nonce variation of the proxies and registrars present in the infrastructure is determined and some users indentities are discovered.

The second dataset corresponding to the second collection hour was used as a test data set. For this case, we also launch SIP attacks against the operational infrastructure. We note that some attacks that are launched during this phase are new; i.e. they are not present in the first data set. The goal of restricting the presence of new attacks in the new data set is to evaluate the efficiency of the detection model towards new attacks.

B. Data pre-processing

Raw *tcpdump* traffic collected from a monitored network is not appropriate for direct usage by the detection models. Therefore, a transformation function, which transforms the raw traffic into attributes records without information and intrusion evidence loss, is used to generate well formed data as input for the detection models. Attributes extraction, as described in Figure 1, summarizes VoIP raw traffic into attributes records. Each SIP signaling flow is transformed into a record composed of different attributes extracted from the raw flows according to the procedure presented in Section III-B. We give in the following paragraphs a thorough description of the two attributes types namely; intrinsic and expert knowledge attributes. Intrinsic attributes are grouped into a class that we call *"first class"* and the other type corresponding to the different attributes computed according to the last flows preceding the current one.

- **First class** This class corresponds to the different attributes that are intrinsic to the VoIP protocol, particularly SIP. Table I presents a non exhaustive list of attributes of this class. We mention that for each flow a time-stamp corresponding to the time of its occurrence is considered to calculate the attributes of the other two classes.
  We mention that the different attributes presented in Table I are intrinsic ones; others are extracted by considering known attacks. As an example, the last three attributes UserName, Nonce and Response are extracted based on the two attacks; nonce variation and password guessing. Therefore, this list is an open one as long as other vulnerabilities and attacks are discovered. Using this list for each flow independently from others is not a good solution. One solution is to find statistical characteristics using the last flows preceding the current one in the near past. This is discussed in the following two classes. We note that the normal flow of the signaling traffic tremendously follows a statistic law as in the different telephony models. Therefore, these following attributes highly contribute to characterizing the normal flow.
- **Second class** The attributes of this class are based on calculating correlation measures between the different flows preceding the current one using the different attributes values indicated in the first class. Figure 6 shows the idea used to compute the corresponding attributes.
  A time window of N seconds (2 seconds for instance) is used for this purpose. These attacks are relevant for VoIP DoS flooding attacks and other attacks that send the same requests with different values such as password guessing or nonce variation. The different attributes of this class are automatically constructed and are summarized into the "Same_To-URI" attributes that examine the flows in the last N seconds that have the same logical recipient as the current flow. We note that the logical originator is not taken into account to calculate the different attributes in order to avoid URI spoofing where an attacker may forge a *"From URI"* header field. However, in a real world, the provider of the service may use ingress filtering and in this case, we may consider the logical initiator of the flow. Since this is not always the case, we do not use it here and consider all possible situations. Table IV presents the

Attribute	Description
Resp_Req	The value of this attribute is "REQUEST" if the considered flow is a request else it is "Response"
SCN	The value of the status code if it is a response (200, 180, etc.) else it is set to "NULL"
Reason_ Phrase	The reason phrase informed from the response (OK, UNAUTHORIZED, etc.)
Method	The value of the method informed from the request (INVITE, REGISTER, etc.)
From_URI	It corresponds to the logical initiator of the request informed in the *"From"* header field
To_URI	This attribute corresponds to the logical recipient informed in the *"To"* header field
From_Tag	The value of the tag parameter informed in the *"From"* header field. It is used to follow a dialog between two UAs
To_Tag	The value of the tag parameter informed in the *"To"* header field. It is used to follow a dialog between two UAs
UserName	This corresponds to the credential value of the username parameter specified in the *"Authorization"* header field
Nonce	It corresponds to the credential value of the nonce parameter specified either in an *"Authorization"* header field or in the *"WWW-Authenticate"* header field
Response	This corresponds to the response parameter specified in the *"Authorization"* header field as a response to the challenge

TABLE I
FIRST CLASS ATTRIBUTES LIST.

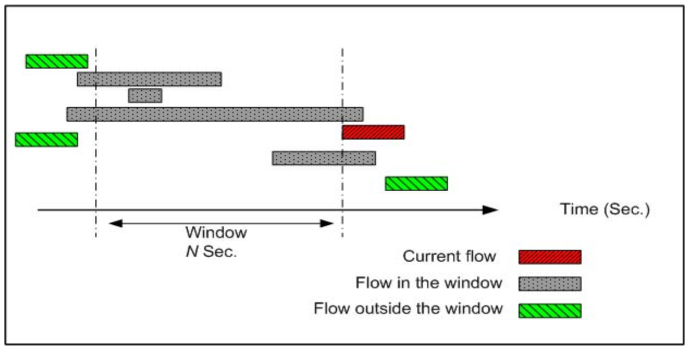

Fig. 6
KNOWLEDGE BASED ATTRIBUTES SPECIFICATION.

different attacks of this class and their descriptions.

- **Third class** A novice attacker may send many requests in a short time window. The second class attributes is sufficient enough to detect the corresponding attack. However, other attackers will take then time and use stealthy techniques to bypass this approach. Therefore, a larger time window to detect these attacks is needed to detect. For this reason, we introduce the third class that considers the last N flows ($N = 200$ for instance) preceding the current one to calculate the corresponding attributes as those in Table IV. The attributes of this class are calculated according to the last N flows preceding the current one. Therefore, we do not report them here.

C. Results

We conduct different experiments over the two data sets presented in Section V-A. We trained our algorithm over the first data set presenting the first collection hour that contains different attack types as those presented in Section IV. We notice that there are new attacks which are only present in the test data set that corresponds to the second hour of collection. These new attacks correspond to the DoS against a client and the nonce variation. Notice also that the DoS against a client is not tested inside the real environment because of the operator constraints. However, the last attack is tested in our local infrastructure. Some rules that are generated automatically from the training data set are given in Table II.

These rules have many advantages in detecting anomalies in signaling flows. Since the rules have the `"IF ... THEN ..."` format, they may be used as a model for a rule based intrusion detection system. Moreover, a VoIP security expert may assess the different rules and can delete or modify some of them if needed.

Using the ruleset generated by the training data set, new flows are examined by checking the different rules for a match. If there is none rule that matches then the flow is considered as new and should be examined to check whether it corresponds to a new attack. If so, we examine the corresponding traffic and if it corresponds to a new attack, we re-inject its traffic in the learning data base to generate its corresponding rule.

The successful detection rate is over 99% by applying the different rules on the training data set. This results means that the different attributes that are determined during the extraction step efficiently characterize the different flows and differentiate between the different attack classes and the normal traffic.

Rule	**Meaning**
Resp_Req $= REQUEST$, same_method_rate $> 72\%$ $- >$class Register_DoS	If the flow crresponds to a request and the percentage of the flows that have the same method request as the current one during the last two seconds is greater than 72% then this flow corresponds to a DoS probably against a registrar
Method$= REGISTER$, diff_username_rate $< 0.1\%$ $- >$class guesspassword	If the method is REGISTER and the diff diff_username_rate is less than 0.1% then this flow is a password guessing attack.
$\vdots$	$\vdots$
Default: New	If none of the rules matches then the current flow corresponds to a new flow and momentarily considered as a new attack.

TABLE II
CLASSIFICATION USING THE DECISION RULES.

Table III gives the different detection rates of the different classes (normal and the different attacks types) in the test data set.

Type	**Old**	**New**
Information gathering	99.25%	Nonce Variation: ($\sim 99.50\%$) detected as DoS
Identity and service theft	99.15%	-
DoS	99.77%	DoS against a client: ($\sim 99.34\%$) detected as New
Normal	99.96%	-

TABLE III
OLD AND NEW ATTACKS DETECTION RATIOS.

The old intrusions correspond to those attacks that are present in the training and test data sets. There only two new attacks that are present only in the test data set. The first is the nonce variation determining attack and the second is the DoS against a client attack. While the nonce variation attack is almost detected as a DoS against a server attack, the occurrences of the DoS against a client attack are detected as a new attack. This latter is detected as new because there is none rule that specifies such a profile with an INVITE method and a high rate of requests during a short time window. However, the nonce variation determining attack category is detected as a DoS against a registrar since this attack category uses the REGISTER request method with a high rate of requests during a short period that corresponds exactly to a DoS using the REGISTER method against a registrar. As a result, we added three attributes related the challenge credentials used in the digest authentication scheme used within the SIP protocol namely; username, nonce and response (see for instance the last attributes mentioned in Table I). To assess our architecture, we re-injected the traffic corresponding to the nonce variation determining into the training data set and new rules are defined for this attack.

We have also used Snort [16] to detect these attacks. We configured it with the latest rules. None of the attacks cited above are detected by Snort since all the packets of the two data sets are well formed and there is none rule in the Snort database that corresponds to any of the attacks cited above. In addition, it is very hard to write the corresponding rules because pattern matching techniques are not appropriate for this kind of attacks.

VI. Conclusion

In this paper, we introduce a framework for detecting anomalies in signaling flows related to the SIP protocol targeting the VoIP networks. The main idea behind our proposal is the attributes extraction from the signaling flows that highly characterize attacks and differentiate between normality and abnormality in a VoIP environment. To take into consideration new VoIP attacks, our mechanism considers new attributes that are relevant for characterizing them. A feedback from new attacks contributes to extend the ability of this framework in detecting other attack variants and new ones.

The different experiments show that our mechanism is successful to detect almost all known attacks and new ones collected in a real testbed.

Our future work includes developing a VoIP alert correlation engine able to detect ongoing attack scenarios that contain successive elementary attacks as those we presented in this paper.

References

[1] The 3rd Generation Partnership Project. Available at: `http://www.3gpp.org/`, 2007.

[2] Y. Bouzida. Principal Component Analysis for Intrusion Detection and Supervised Learning for New Attack Detection. PhD Thesis, March 2006.

[3] Y. Bouzida and F. Cuppens. Detecting Known and Novel Network Intrusions. In 21^{st} *IFIP International Information Security Conference (SEC'2006)*, Karlstad, Sweden, May 2006.

[4] U. M. Fayyad. on the induction of decision trees for multiple concept learning. PhD Thesis, 1991.

[5] J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, and L. Stewart. HTTP Authentication: Basic and Digest Access Authentication, RFC 2617. Available at: `http://www.ietf.org/rfc/rfc2617.txt`, June 1999.

[6] KDD Cup 99 Intrusion Detection Datasets. Available at: `http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html`, 1999.

[7] W. Lee, S. J. Stolfo, and K. Mok. A Data Mining Framework for Building Intrusion Detection Models. In *Proceedings of the 1999 IEEE Symposium on Security and Privacy (SSP'99)*, May 1999.

[8] T. M. Mitchell. *Machine Learning*. McGraw Hill, 1997.

[9] M. Nassar, R. State, and O. Festor. Intrusion detection mechanisms for VoIP applications. Available at: `http://www.iptel.org/voipsecurity/workshop/program_1stjune2006.php`, June 2006.

[10] V. Paxson. Bro: A system for detecting network intruders in real-time. *International Journal of Computer and Telecommunications Networking Archive*, 31(23-24), December 1999.

[11] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1:1–106, 1986.

[12] M. Roesch. Snort - Lightweight Intrusion Detection for Networks. In *13th Systems Administration Conference - LISA 99*, 1999.

[13] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: Session Initiation Protocol, RFC 3261. Available at: `http://www.ietf.org/rfc/rfc3261.txt`, June 2002.

[14] H. Schulzrinne, P. S. Casner, R. Frederick, and V. Jacobson. RTP: A Transport Protocol for Real-Time Applications, RFC 1889. Available at: `http://www.ietf.org/rfc/rfc1889.txt`, January 1996.

[15] H. Sengar, D. Wijesekera, H. Wang, and S. Jajodia. VoIP Intrusion Detection Through Interacting Protocol State Machines. In *Proceedings of the 2006 International Conference on Dependable Systems and Networks (DSN'06)*, Pennsylvania, USA, June 2006.

[16] Snort Network Intrusion Detection System. Available at: `http://www.snort.org`, 2007.

[17] A. Valdes and K. Skinner. Adaptive, Model-Based Monitoring for Cyber Attack Detection. In *Recent Advances in Intrusion Detection, 5th International Symposium, RAID 2000*, pages 80–92, Toulouse, France, October 2000.

[18] Y. S. Wu, S. Bagchi, S. Garg, N. Singh, and T. K. Tsai. SCIDIVE: A Stateful and Cross Protocol Intrusion Detection Architecture for Voice-over-IP Environments. In *Proceedings of the 2004 International Conference on Dependable Systems and Networks (DSN'04)*, Florence, Italy, July 2004.

Feature	Meaning
-count "same_To-URI"	Number of flows to the same URI as the current one in the past N seconds
The following features refer to these flows with the same To-URI value during the past N Seconds	
same_ method_ rate	Percentage of the "count" flows that have the same method value
Diff_ meth_ rate	Percentage of the "count" flows that have different methods
same_QS_rate	Percentage of the "count" flows that have the same Resp_Req value
diff_QS_rate	Percentage of the "count" flows that have different Resp_Req values
same_scn_rate	Percentage of the "count" flows that have the same status code value
diff_scn_rate	Percentage of the "count" flows that have different status code values
same_rp_rate	Percentage of the "count" flows that have the same reason phrase value
diff_rp_rate	Percentage of the "count" flows that have different reason phrase values
same_username_rate	Percentage of the "count" flows that have the same username value
diff_username_rate	Percentage of the "count" flows that have different username values
same_nonce_rate	Percentage of the "count" flows that have the same nonce value
diff_nonce_rate	Percentage of the "count" flows that have different nonce values
same_response_rate	Percentage of the "count" flows that have the same response value
diff_response_rate	Percentage of the "count" flows that have different response values
-method_count	Number of flows that have the same Method as the current one in the past N seconds
The following features refer to these flows with the same Method value during the past N Seconds	
meth_same_To-URI	Percentage of the "method_count" flows that have the same To-URI value
meth_diff_To-URI	Percentage of the "method_count" flows that have different To-URI values
-QS_count	Number of flows t that have the same Resp_Req as the current one in the past N seconds
The following features refer to these flows with the same Resp_Req value during the past N Seconds	
QS _same_To-URI	Percentage of the "QS_count" flows that have the same To-URI value
QS_diff_To-URI	Percentage of the "QS_count" flows that have different To-URI values
-scn_count	Number of flows that have the same status code as the current one in the past N seconds
The following features refer to these flows with the same status code value during the past N Seconds	
scn_same_To-URI	Percentage of the "scn_count" flows that have the same To-URI value
scn_diff_To-URI	Percentage of the "scn_count" flows that have different To-URI values
-rp_count	Number of flows that have the same Reason Phrase as the current one in the past N seconds
The following features refer to these flows with the same reason phrase value during the past N Seconds	
rp_same_To-URI	Percentage of the "rp_count" flows that have the same To-URI value
scn_diff_To-URI	Percentage of the "rp_count" flows that have different To-URI values
-username_count	Number of flows that have the same username as the current one in the past N seconds
The following features refer to these flows with the same username value during the past N Seconds	
username_same_To-URI	Percentage of the "username_count" flows that have the same To-URI value
username_diff_To-URI	Percentage of the "username_count" flows that have different To-URI values
-nonce_count	Number of flows that have the same nonce as the current one in the past N seconds
The following features refer to these flows with the same nonce value during the past N Seconds	
nonce_same_To-URI	Percentage of the "nonce_count" flows that have the same To-URI value
nonce_diff_To-URI	Percentage of the "nonce_count" flows that have different To-URI values
-response_count	Number of flows that have the same response as the current one in the past N seconds
The following features refer to these flows with the same response value during the past N Seconds	
response_same_To-URI	Percentage of the "response_count" flows that have the same To-URI value
response_diff_To-URI	Percentage of the "response_count" flows that have different To-URI values

TABLE IV
SECOND CLASS ATTRIBUTES LIST.