Detecting Denial of Service Message Flooding Attacks in SIP based Services

Zoha Asgharian[i]; Hassan Asgharian[ii*]; Ahmad Akbari[iii] and Bijan Raahemi[iv]

ABSTRACT

Increasing the popularity of SIP based services (VoIP, IPTV, IMS infrastructure) lead to concerns about its security. The main signaling protocol of next generation networks and VoIP systems is Session Initiation Protocol (SIP). Inherent vulnerabilities of SIP, misconfiguration of its related components and also its implementation deficiencies cause some security concerns in SIP based infrastructures. New attacks are developed that target directly the underlying SIP protocol in these related SIP setups. To detect such kinds of attacks we combined anomaly-based and specification-based intrusion detection techniques. We took advantages of the SIP state machine concept (according to RFC 3261) in our proposed solution. We also built and configured a real test-bed for SIP based services to generate normal and assumed attack traffics. We validated and evaluated our intrusion detection system with the dump traffic of this real test-bed and we also used another specific available dataset to have a more comprehensive evaluation. The experimental results show that our approach is effective in classifying normal and anomaly traffic in different situations. The Receiver Operating Characteristic (ROC) analysis is applied on final extracted results to select the working point of our system (set related thresholds).

KEYWORDS

Denial of Service, Session Initiation Protocol, Flooding Attacks, State Machine, Intrusion detection system

1. INTRODUCTION

The Session Initiation Protocol (SIP) is an application layer protocol standardized by the Internet Engineering Task Force (IETF), and is used for creating, modifying and terminating sessions [1, 2]. SIP is structured as a layered protocol, which means that its behavior is described in terms of a set of quite independent processing stages with only a loose coupling between each stage [1, 2].

The lowest layer of SIP is its syntax and encoding and the second layer is the transport layer. It defines how a client sends requests and receives responses, and also, how a server receives requests and sends responses over the network. The third layer is the transaction layer. Transactions are the fundamental component of SIP. A transaction is a request sent by the client transaction layer to the server transaction layer, along with all responses to that request which are sent from the server transaction layer back to the client transaction layer. The transaction layer handles application-layer re-transmissions, matching of responses to requests, and application-layer timeouts. The layer above the transaction layer is called the transaction user (TU). When a TU wants to send a request, it creates a client transaction instance and passes the request along with the destination IP address, its port, and its transport layer information (Figure 1).

[i] Zoha Asgharian graduated from computer engineering school of Iran University of Science and Technology, Tehran, Iran (email: z_asgharian@comp.iust.ac.ir)

[ii] * Corresponding Author, Hassan Asgharian is PhD student in computer engineering school of Iran University of Science and Technology, Tehran, Iran (email: asgharian@iust.ac.ir)

[iii] Ahmad Akbari is an associate professor in the computer engineering school of Iran University of Science and Technology, Tehran, Iran (email: Akbari@iust.ac.ir)

[iv] Bijan Raahemi is with University of Ottawa, Canada (email: raahemi@iust.ac.ir)

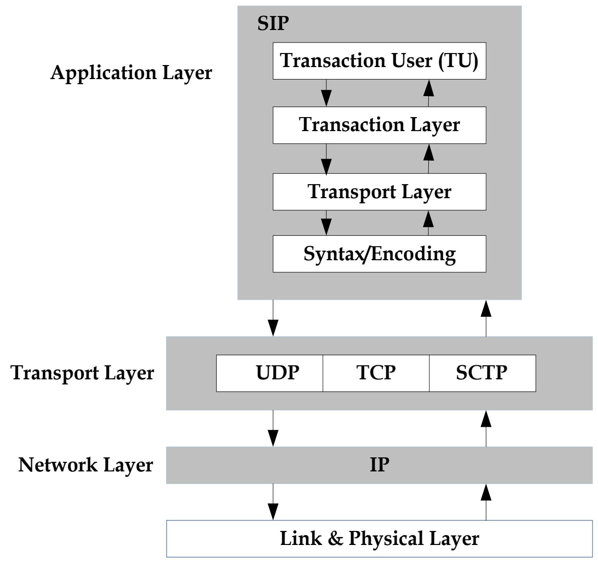

Figure 1: SIP protocol layers

In 2005, the US National Institute of Standards and Technology declared DoS attacks to be a serious threat for SIP infrastructures [3]. A DoS attack makes particular network resources unavailable by flooding it with illegitimate packets to seize its bandwidth, memory and CPU. DoS attacks can be classified as illustrated in Figure 2. The attacks categorized into two broad groups: intentional attacks and non-intentional attacks. Non-intentional attacks are usually the result of implementation shortcomings or architecture misconfigurations. However, intentional attacks are initiated purposefully by intruders. Intentional attacks can be further subdivided into flooding and protocol misuse attacks. Flooding attacks are also referred to as exhaustion or depletion attacks because of their final goal of depleting one or more resources of the victim and making it incapable of conducting its regular tasks and processing the incoming requests [4].

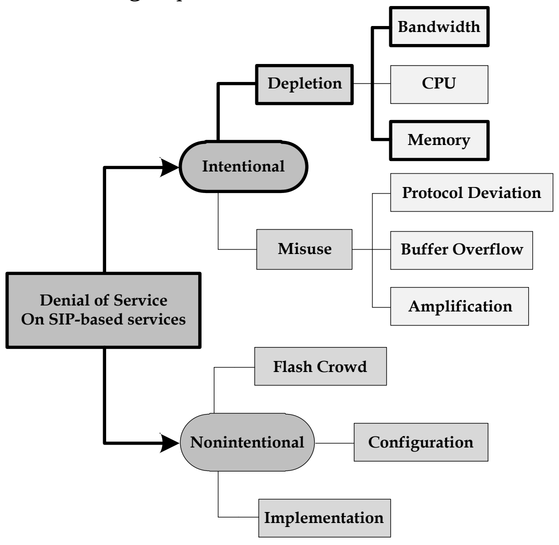

Figure 2: Classification of DoS Attack on SIP-based services

This paper focuses on detecting some of the important intentional attacks that are planned to deplete the victim's resources as indicated in Figure 2. The attack scenarios considered in the paper are explained below.

(I) Bandwidth Depletion Attacks: In this scenario, the attacker generates a large number of SIP requests (e.g. INVITE packets) and sends them in a short period of time to the SIP server to deplete its bandwidth. There is no difference and accuracy in parameter settings of this attack. Although this scenario is very similar to the overload state of the SIP proxy servers but it is different from them because its target is to deplete the bandwidth of the SIP servers and therefore the validity of generated SIP messages becomes unimportant. This scenario is illustrated in the Figure 3.

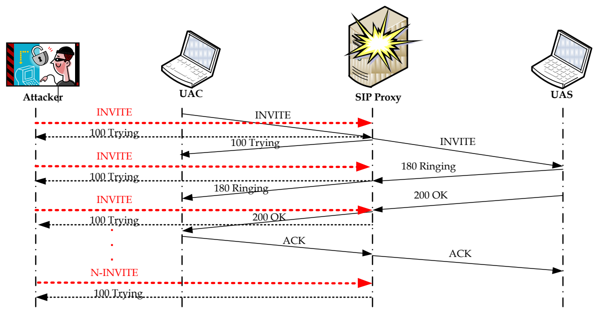

Figure 3: INVITE flooding Attack

(II) Brute Force Attacks: The generation process of this attack is similar to the basic INVITE flooding explained in the previous section with two main differences: target and parameters. The focus of intruder in this attack is SIP proxy's main memory. The attacker misuses the SIP session management sequence to seize the memory of the proxy server. There are two different definitions in SIP for sessions: transaction and dialog. These sessions are defined in the application layer and based on the related features of the SIP traffic. A SIP transaction consists of a request message with all of its related responses. It is identified by *VIA Branch* and *CSeq Method* of the SIP header fields. A set of related SIP transactions make a SIP dialog. A SIP dialog is identified by three SIP parameters: *Call-id*, *To Tag* and *From Tag*. Figure 4 shows these definitions in the typical SIP signaling process. Most of the SIP proxy servers are stateful and work on transaction level and devote memory to each SIP request. The goal of intruder in this attack is to deplete the memory of SIP proxy server and for this reason the traffic is generated with different session parameters. If the number of generated SIP requests in a period of time is more than a specified numbers which is computable, the proxy server will eventually become unavailable. For instance, the simplest approach for mounting an attack on the memory of a SIP server is to initiate a large number of SIP sessions with different session identities (i.e. different *VIA Branch* and *CSeq Method* fields). If the number of generated calls and transactions is greater than the threshold for which the memory of the SIP proxy was designed, then the SIP server will start dropping the incoming requests.

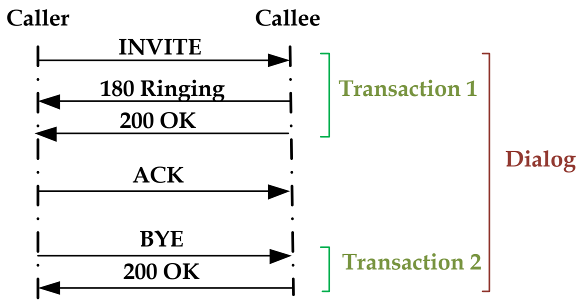

Figure 4: SIP session definition (Transaction vs. Dialog)

(III) Incomplete Transactions with Host Cooperation (Ringing based attacks): By sending provisional responses, e.g. 1xx replies, to INVITE requests, a receiving user agent can prolong the lifetime of a transaction to several minutes. In this attack scenario, the attacker sends INVITE requests to destinations that reply with 1xx messages but do not send final responses. With this approach, the attacker needs to send even fewer requests to deplete the memory of the SIP server. For further prolonging the lifetime of the transactions, the used destinations have to cooperate and reply with provisional responses to the received requests [4] (Figure 5).

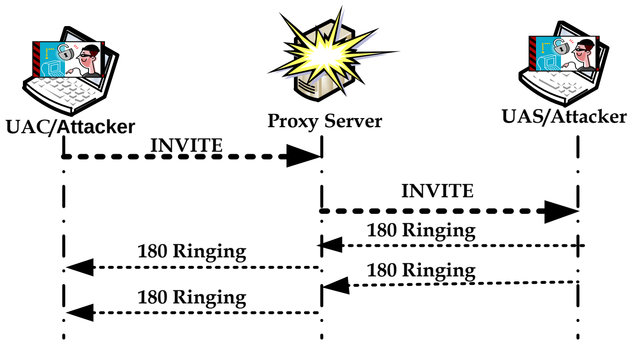

Figure 5: Incomplete transactions with cooperating hosts

(IV) Misusing SIP Authentication (REGISTER Flooding): The default authentication mechanism of SIP is HTTP digest authentication. The authenticating server generates a nonce and adds it to a 401 Unauthorized or 407 Proxy Authentication Required responses in a WWW-Authenticate or Proxy-Authenticate header. The user agent then ignores the first transaction and generates a new request with its credentials calculated based on the shared secret with the provider and the nonce. While generating the 407 response and receiving the new request from the user, the server usually maintains a copy of the nonce. Attacker generates a request and includes it in the FROM header the identity of a subscriber of the attacked proxy or registrar server. Once asked to verify his/her identity, the attacker just ignores this challenge and starts a new request with another session identity. The attacked server keeps the allocated memory for a certain period of time. The memory of the attacked server is, hence, consumed by the saved nonce and the transaction data. This comes in addition to the wasted CPU resources for calculating the nonce [4] (Figure 6).

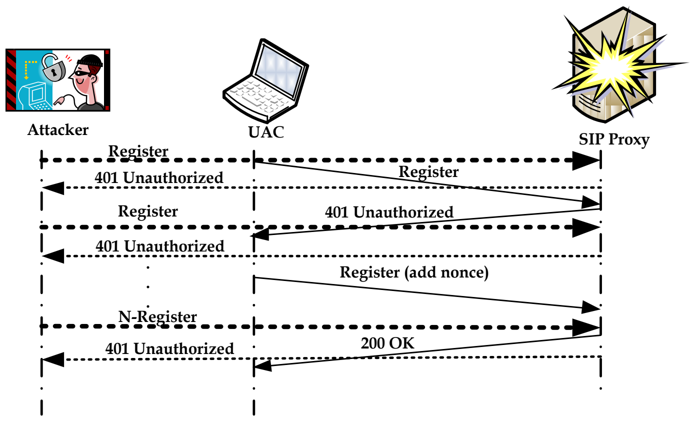

Figure 6: REGISTER flooding attack

The main contribution of this paper is to propose an intrusion detection mechanism based on the SIP state machine in order to detect the SIP DoS attacks. The proposed solution is scalable due to it is located in front of SIP proxy servers and in the ingress point of network and it does not care about the infrastructure. It is also extendable because its architecture is inspired from SOC main components and can be extend easily to detect other types of SIP based attacks. We also proposed a sample response system based on the black list idea in the proposed system. The other main contribution of this paper is the test-bed proposed for normal traffic generations and SIP related attacks upon which its results can be used as an evaluation dataset for comparing different related Intrusion Detection Systems.

The rest of this paper is organized as follows. Section 2 briefly reviews the related works. In Section 3, we discuss our proposed framework for SIP intrusion detection. The architecture of our test-bed is explained in section 4 and the experiment setup and result analysis are presented in section 5. Finally, we conclude our paper with an outlook to the future research.

2. Related Work

A two-layer DoS prevention architecture for SIP is proposed in [23] and [3]. The first layer is comprised of a bastion host that protects against well-known network-layer attacks (such as TCP SYN flooding) and SIP-flooding attacks. The second layer is located at the SIP proxy, and is composed of modules that perform signature-based detection of malformed SIP messages and a non-blocking DNS cache to protect against attacks involving SIP URIs with irresolvable DNS names. The authors of [23] conducted a series of evaluations in an experimental test-bed where they validated the effectiveness of their architecture to block or mitigate a number of DoS attacks. The similar system is proposed in [6] that experimentally evaluate a specification-based intrusion-detection system for denial of service attacks. In some other related works, some systems are proposed based on using filtering techniques like Bloom filters [11] to detect messages that are part of a denial of service attack in SIP by determining the normal number of pending sessions for a given system and configuration based on profiling. Authors in [25] described the design and implementation of a SIP-aware, rule-based application-layer firewall that can handle denial of service in the signaling and media protocols. They use hardware acceleration for the rule matching component, allowing them to achieving filtering rates on the order of hundreds of transactions per second which is not applicable in all networks and limit the usage of their proposed system to their platform. They also experimentally evaluate and validate the behavior of their prototype with a distributed test-bed involving synthetic benign and attack traffic generation. Using cross layer techniques for intrusion detection is another approach in SIP domain. In [7] a special rule based intrusion detection system was designed for correlating the events of media and control layers to detect the anomalies of VoIP systems. The much focus of some previous studies like [9] was on the high volume DoS attacks. The authors of [9] presented an open architecture for monitoring SIP traffic. Their key design goals (like all other studies) include transparency, scalability, extensibility, speed and autonomous operations. Although the infrastructure of IMS adds some constraints on its services but the underlying control protocol of it is also SIP. Therefore some of previous studies like [12] conducted an analysis of three anomaly detection algorithms for detecting flood attacks in IMS: adaptive threshold, cumulative sum, and Hellinger distance. They use synthetic traffic data to determine the detection accuracy of these algorithms in the context of a SIP server being flooded with SIP messages. The other related class of papers for detecting SIP DoS attacks focused on the SIP state machine [5]. The author of [5] proposed a method to detect INVITE flooding attacks, where the primary finite-state machine of the SIP transactions are modified in such a way those anomalies can be detected in a stateful manner. We share the same idea with them in using SIP state machine with one fundamental difference that we do not change the original SIP state machine. In [6], the authors proposed a specification-based intrusion detection framework based on the SIP finite-state machine to distinguish deviations from the normal or expected behavior. They also proposed a method for mitigation of detected attacks. In spite of efforts done in detecting intrusions on SIP based applications [10, 11, 12], there is no common comprehensive labeled dataset in this field. The only published dataset in this field is [21] that have many shortcomings and specified to only basic INVITE flooding. We present a real test-bed to prepare a labeled dataset which is described briefly in this paper and its details are given in [19].

3. The Proposed Intrusion Detection System

The main components of the proposed intrusion detection system architecture are inspired from the Security Operation Center (SOC) which is made up of five distinct modules: *event generators, event collectors, message database, analysis engines and reaction management software* [13]. The architecture of our system is shown in Figure 7.

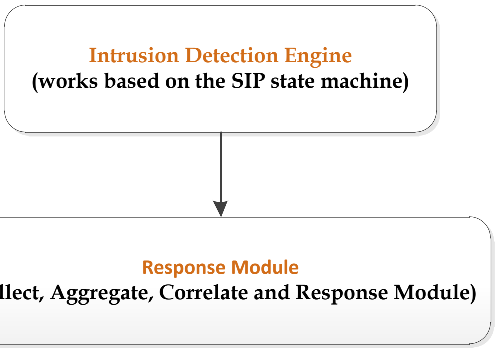
Intrusion Detection Engine
(works based on the SIP state machine)
Response Module
(Collect, Aggregate, Correlate and Response Module)

Figure 7: Main architecture of proposed system

Below we explain the relevance of our presented modules with those defined in the SOC architecture:

(1) **Event generators**, the detection module (which will be explained in the next section) acts as the event generator in the proposed framework which is our main contribution.
(2) **Event collectors and message database**, all important generated events are handled in simple data structures and databases.
(3) **Analysis engines,** in this module, an anomaly score is computed based on the database records. The anomaly score is a function of the user's activities which is

calculated by the generated events of the detection engine. This computed score is compared with some predefined thresholds and returns the traffic status.

(4) **Reaction management components**, after recognizing the traffic status, the detected intruders are added to the black list and the system limits their access with some policies. The intruders are identified by the main attributes of the SIP header. The details of the reaction module are explained in next section.

A. *Detection Mechanism*

Holding the status of active and in-progress transactions in the SIP server consumes both memory and CPU processing resources. Therefore, attackers can degrade performance of the SIP server using the following two approaches:

(1) Send a large number of transactions to the server within a short time to deplete its memory, bandwidth and CPU.

(2) Extend the time of each transaction in the SIP server to seize its memory gradually.

Intrusion detection approaches can be divided into misuse detection, anomaly detection and specification-based detection [14]. In this terminology, our approach is a combination of both specification-based and anomaly-based detection techniques. We combined the attributes of the SIP messages with information about statistics that need to be maintained to detect anomalies. Therefore, our proposed approach mitigates the weaknesses of the two approaches while magnifying their strengths. We considered the SIP specific features (SIP transaction and SIP dialog) that helped us to have a better decision about the current status of the traffic. We also considered the SIP state machine transitions in our detection engine for more accuracy and used the idea of SOC to make a complete security framework for SIP intrusion detection and response. The state machine of the proposed detection engine is shown in Figure 8. The illustrated states are derived from the definition of transactions in SIP messages, and transitions are formed according to different headers of the incoming SIP messages. Each packet raises an event in the state machine. In each state, with entrance of a new packet, our security engine extracts the required fields of the SIP header, and based on its SIP method and transaction identity fields (*CSeq Method* and *VIA Branch*) makes a decision about next state (transition). INVITE and REGISTER are two main methods in SIP because almost all of the SIP entities should answer the INVITE and the registration process should be repeated periodically. Each transaction adds a new entry to the status table which keep track of SIP transactions, while its respected transaction response, deletes an appropriate line from this table. We prespecified three thresholds based on the size of this table for distinguishing the status of input traffic (these values are shown as Th_x in Figure 8). The procedure of setting the threshold will be discussed in next section.

As shown in Figure 8, depending on the input SIP message, the detection system starts adding their appropriate fields to its status table to keep the system state. The status table is updated by each entrance of response messages. The state transition table is shown in the Table 1. In the abovementioned state machine, we keep track the number of valid SIP transactions which should not be more than the specified value. In other words, if some transactions exist in the system more than the normal period (according to RFC 3261, this time is about 32 seconds), the appropriate alarm should be raised. These three thresholds help to detect different attacks. For instance, in the case of INVITE flooding, the system will remain in the state A and the number of in-progress transactions will become higher than the predefined threshold which leads to raising alarm.

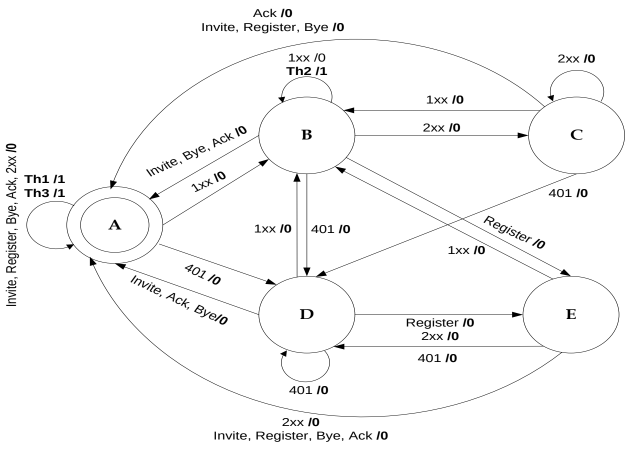

Figure 8: Proposed state machine for flooding attack detection (INVITE, REGISTER, RINGING)

TABLE 1
STATE TRANSITION TABLE

Event	Next State\Output									
Current State	INVITE	Register	Bye	ACK	1xx	2xx	401	3xx, 5xx, 6xx	Th1	Th2	Th3
A	A/0	A/0	A/0	A/0	B/0	A/0	D/0	A/0	A/1	-	A/1
B	A/0	E/0	A/0	A/0	B/0	C/0	D/0	A/0	-	B/1	-
C	A/0	A/0	A/0	A/0	B/0	C/0	D/0	A/0	-	-	-
D	A/0	E/0	A/0	A/0	B/0	E/0	D/0	A/0	-	-	-
E	A/0	A/0	A/0	A/0	B/0	A/0	D/0	A/0	-	-	A/1

B. *Reaction Mechansim*

We equip our security engine with a report module. After detecting new intrusion, we add specific information about it (SIP URI: FROM, TO) to the run-time hash table (used for black list), and also append the alert log file with appropriate notifications. Using the terminology of response systems, this approach belongs to the notification and prevention categories. The alert generation can be considered as a notification response which the administrator can use to take a proper action. The black list can be used to prevent the access of intruders temporally. It works independently from any existing firewalls in the system.

In next section the evaluation of the proposed system is described and the experimental setup is explained.

4. LABELED DATASET

In this section, the configuration of our test-bed, traffic generation models and the selected features of SIP packets used in our proposed framework, are described in the following different subsections. More detailed information on this labeled dataset can be found in [19].

4.1 Test-bed Configuration

A test-bed is a platform for experimentation of large development projects. Test-beds allow for rigorous, transparent, and replicable testing of scientific theories, computational tools, and new technologies. A typical test-bed could include software, hardware, and networking components. In Figure 9 the main components and architecture of our test-bed is shown.

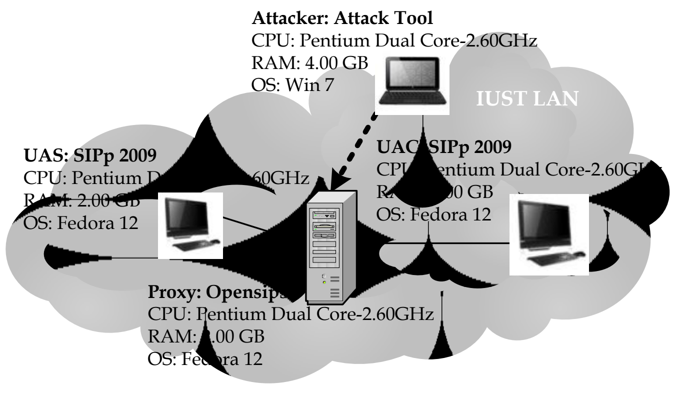

Figure 9: Our Test Bed

Modeling the behavior of normal users, we used SIPp [15] which is a well-known open-source SIP traffic generator. We also configured OPENSIPS [20] as our proxy server in our test-bed. The details and parameters of our traffic generation model is described in the following sections.

4.2 Traffic Generation Model

The traffic generation model consists of two separate parts for normal and attack conditions. In this section, the generated normal traffic is described first and after that the properties of our attack tool and its generated traffic is defined. We used SIPp for normal traffic generation. We configured the test-bed shown in Figure 9 with more than one instance of this tool to aggregate normal traffic patterns. Some of the important parameters of normal traffic are as below:

(1) **Probabilistic distribution of call duration:** Since the number of normal users is very large in real telephone networks, the probabilistic distribution of them seems low-importance at most times. However, we used two well-known distributions which can be important in low volume traffics. We set probabilistic distribution type of call duration to exponential and normal distributions.

(2) **Calls per second:** This parameter has direct relationship with the number of active users, which is a fraction of the maximum number of users. Therefore, we assumed three types of networks (low-, medium- and high-volume traffic) with different number of users and simulated their normal behaviors. We derived the call per second based on the number of users and the activity parameter of SIP network. The following equation describes this relationship:

$$\alpha \times N = \lambda \times T \rightarrow \lambda = \frac{\alpha \times N}{T} \qquad (1)$$

Where α is the percentage of active users in the network which is selected based on the normal status of the network and N represents the maximum number of users which is a known parameter in each network and T is call duration that is described later and λ is the call per second ratio calculated based on these parameters.

(3) **Call duration:** The other related important parameter of normal traffic is call duration. We consider similar assumptions to mobile telephone network for setting the call duration and assumed that more than 50% of active users (active calls) are completed in less than 60 seconds and about 20% of them are lasted more than 300 seconds and the others are finished between these 60 to 300 seconds. For normal traffic generation we can take into consideration two approaches: I) generate a traffic with normal distribution with large standard deviation and average equals to 300 seconds or II) generate three separates traffic flows based on the appropriate parameters and aggregate their outputs. We chose the second approach. In addition to these active requests (INVITE, BYE, ACK, CANCEL) and their respected responses (1xx to 6xx), the REGISTER is another important SIP request which should be generated periodically for each user. The period of REGISTER requests were considered almost 3600 seconds.

(4) **Random selection of SIP message fields**: It is necessary to produce SIP packets with different SIP URIs (different FROM and TO fields in SIP message) and also make different call flows (dissimilar SIP dialogs and transactions). This assumption was considered in our test-bed to make its output more reliable.

According to the above parameters, the normal traffic has 10, 18 and 82 call per seconds for low, medium and high volume traffic, respectively (the network activity ratio (α) is set to 0.05 in all of these scenarios). More details about normal traffic can be found in [19].

The attacker entity is implemented as an independent bot run on separate machine. This bot is implemented in C# and has capability to launch different considered DoS attacks on a target (SIP proxy server). More details about this attack tool are available on [17] and [19]. The final aggregated traffic which used as our dataset comes from the generated traffic of this attack tool and the generated normal traffic. The considered SIP DoS message flooding attacks in our dataset are: basic request flooding (bandwidth depletion), brute force message flooding (memory depletion) and ringing based attacks (memory depletion).

4.3 Selected Features for Labeled Dataset

Described in previous sections, our intrusion detection system is implemented based on the SIP state machine. For this reason the selected features should represent the SIP transactions and in some case SIP dialogs. So, the *VIA Branch* and *CSeq Method* fields are selected for distinguishing transactions and *Call-id*, *To Tag* and *From Tag* are extracted as dialog identifier. All required parameters of the proposed detection engine are extracted explicitly from SIP header fields. The required header fields are abstracted in Table 2.

TABLE 2
REQUIRED HEADER FIELDS OF SIP FOR DETECTION SYSTEM

Attribute	Description
Method, CSeq Method	It indicates the type of the SIP message and its related transaction.
From, From TAG	It corresponds to the logical initiator of the request and its random selected Tag which are derived from the FROM header field.
To, To Tag	It corresponds to the logical recipient of the request and its random selected Tag which are derived from the TO header field. It is used to follow a dialog between two UAs.
VIA Branch	Via field in SIP header contains a branch parameter that is unique in each transaction.
Call-ID	It contains a globally unique identifier for each dialog. It is generated by the combination of a random string and the softphone's host name or IP address. This feature is derived from Call-ID field in SIP header.

4.4 Prepared Dataset

In previous sections, we introduced our test-bed setup process. We configured a set of normal and attack scenarios in this platform. The configuration selections were constrained by ensuring a relatively short time for each experiment (between 5 and 20 minutes) and a supportable overload for SIP server machine (our SIP server has the capability to respond all calls during execution of test scenarios). The resulting traces of all experiments are available on [17]. While we knew the IP address of the attacker nodes in our test-bed, we were able to label each packet in the final dataset. Although the experimental setup of the test-bed is not perfect in this paper but we prepared a platform for evaluating the intrusion detection systems which can be used for evaluation of different detection mechanisms. In this paper, some of the related information of the prepared dataset is described and more information about the dataset and related topics can be found in [17], [18] and [19].Table 3 summarizes some of the related traffic used in this paper.

5. EVALUATION RESULTS

The proposed detection engine is located independently, in the ingress point of the SIP network. Its behavior is completely transparent to any users. In other words, our intrusion detection system receives all input and output traffic of proxy server and makes decision about the current traffic status. If we want to add the response module to this system (e.g. the described simple blacklist), the proposed system can filter the input traffic of proxy server. The functional block diagram of our intrusion detection system is shown in Figure 10. The implementation is done in C#.

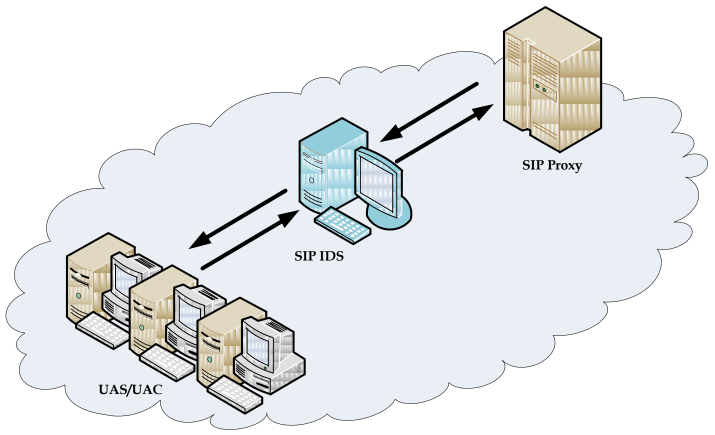

Figure 10: Location of our proposed SIP intrusion detection system

TABLE 3
SUMMARY OF PREPARED LABELED DATASET

Name	Attack Type	Attack Intensity (pps)	Attack Applied Time (second)	Trace Duration (min)	Normal Active Calls (cps)
S1,S2, S3	None	None	None	5	10, 18, 82
S4, S5, S6	INVITE	1000	Random	5	10, 18, 82
		500*			
	REGISTER	200			
	RINGING	50			
S7, S8, S9	INVITE	1000	Random	10	10, 18, 82
		500*			
	REGISTER	200			
	RINGING	50			
S10,S11,S12	INVITE	1000	Random	20	10, 18, 82
		500*			
	REGISTER	200			
	RINGING	50			
S13	INVITE	1000	Random	5	10
		500*			
S14	REGISTER	200	Random	5	10
S15	RINGING	50	Random	5	10

* Brute force INVITE flooding (scenario II)

As we explained in previous sections, our proposed SIP security engine needs three thresholds. For obtaining the best value for these thresholds, the ROC (Receiver Operating Characteristic) analysis was used. We repeated the experiments on labeled dataset with different values. The best values were selected on the short scenario (S4, S5, S6), and the derived thresholds were used in the next experiments (S7 to S15). Figure 11 shows one sample ROC graph.

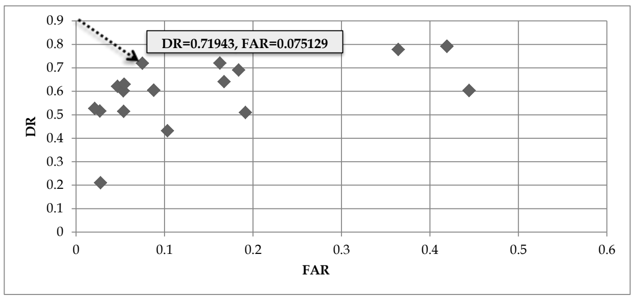

Figure 11. Sample of our ROC graph for dataset 4

The detection rates and false alarms were calculated using the following equations:

$$Detection\ Rate\ (DR) = \frac{True\ Positive}{True\ Positive + False\ Negative} \qquad (2)$$

$$False\ Alarm\ Rate\ (FAR) = \frac{False\ Positive}{False\ Positive + True\ Negative} \qquad (3)$$

After obtaining and setting the working point of system based on the ROC graph (threshold values), the results of applying these thresholds on other scenarios are summarized in Table 4.

TABLE 4
SUMMARY OF OUR SIP SECURITY ENGINE EVALUATION

Scenario Name	DR	FAR
S7	0. 772	0.090
S8	0.869	0. 015
S9	0.689	0.073
S10	0.869	0.117
S11	0.879	0.034
S12	0.732	0.099
S13	0.828	0.155
S14	0.998	0.020
S15	0.981	0.026
Average	0.856	0.077

If the input traffic does not follow the normal distribution, we will see some periods of burst traffic. For our purpose, the effects of this phenomenon on the proposed system, we generated some extra traffics with short time and high rates. The specification of our generated traffic and its related results were reported on [19]. In comparison with the normal traffic rate, our assumptions in generating burst traffics are pessimistic.

TABLE 5
BURST TRAFFIC SPECIFICATION AND ITS RESULTS

Traffic Spec.	DR	FAR
10 cps, 10 min	0.7719	0.0900
200 cps, 20 sec (repeated each 100 sec)	0.7499	0.2126
100 cps, 20 sec (repeated each 100 sec)	0.8666	0.1910
82 cps, 20 sec (repeated each 100 sec)	0.7737	0.0672
50 cps, 20 sec (repeated each 100 sec)	0.8297	0.0662
Average	0.79836	0.1254

As shown in Table 5, the false alarm rate as expected increases when the burst traffics were added to the generated traffics (it means that our solution labeled some benign traffic as an attack).

In order to evaluate the effectiveness of the proposed detection system, we repeated these experiments on another available dataset [21]. Although this labeled dataset has focused only on the SIP invite flooding attacks (basic message flooding), but it can help us to evaluate the presented intrusion detection system. Therefore we applied our algorithm on this dataset and the results are summarized in Table 6. The authors of [21] did not report any measurable results that we can have any comparison with their results. This dataset was collected with two different well-known SIP proxy servers (OPENSIPS and ASTERISK) with the same configuration.

TABLE 6
RESULT OF APPLYING THE PROPOSED SYSTEM ON [21]

Scenario	DR	FAR
iflood-1-opensips-100	0.53751	0.29821
iflood-10-opensips-100	0.87953	0.15712
iflood-100-opensips-100	0.9088	0.07658
iflood-1000-opensips-100	0.89647	0.10847
iflood-1-asterisk-10	0.71731	0.08123
iflood-10-asterisk-10	0.96358	0.16424
iflood-100-asterisk-10	0.98487	0.07487
iflood-1000-asterisk-10	0.9635	0.25743
Average	0.85645	0.15227

The reported detection rates and false alarm values of Table 6 are calculated based on the one working point of the ROC graph. For instance, the ROC graph for "*iflood-100-opensips-100*" and "*iflood-100-asterisk-10*" soiranecs rae shown in Figure 12 and Figure 13 levitcepsery .

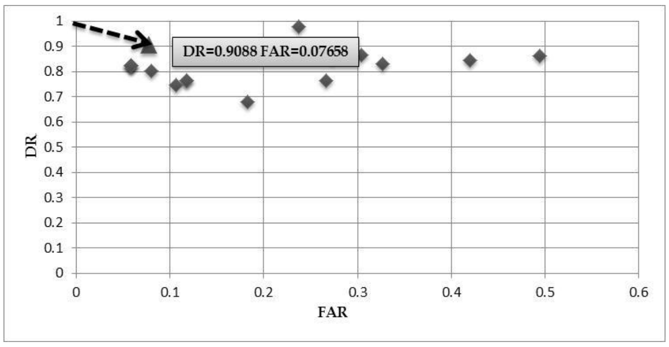

Figure 12: ROC graph of "*iflood-100-opensips-100*" from Table6

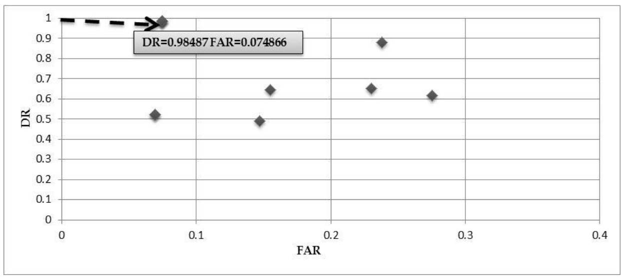

Figure 13: ROC graph of "*iflood-100-asterisk-10*" from Table 6

The experimental results on two different datasets show that the proposed system is capable of detecting different types of DoS message flooding attacks in SIP based services. Although our experiment setup is based on VoIP though we didn't have any assumption on this application and the proposed solution can be used in other SIP based systems.

6. Conclusion and Future Works

We presented a framework for securing the SIP proxy servers in this paper. The proposed system consists of two separate modules for intrusion detection and intrusion response. Although we focused more accurately on intrusion detection part of it in this paper, we also proposed a simple blacklist approach in application layer to disable the access of intruders to SIP proxy servers. Our proposed detection system is formed based on the SIP state machine by considering the special definition of dialogs and transactions in SIP and the components of the response system are inspired from SOC architecture. We plan to enhance our response part of the proposed system in our future works. We considered the scalability, robustness and also performance against tests in the process of design and implementation of the framework. In addition to designing the security framework, we prepared a SIP dataset for the evaluation (and also comparison) of intrusion detection systems and make it publically available in [17]. We plan to add the other SIP attacks especially BYE and CANCEL attacks to our framework in near future and we also continue our work on SIP response and detection systems to have a comprehensive SIP security system. We also focus on other SIP related intrusions like SIP billing attacks and SIP CPU based attacks.

7. References

[1] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Spark, M. Handley, and E. Schooler. "Session Initiation Protocol", 2002. RFC 3261.
[2] A. Ahson, M. Ilyas. "SIP Handbook", Taylor & Francis Group, 2009.
[3] S. Ehlert, D. Geneiatakis, T. Magedanz. "Survey of network security systems to counter SIP-based denial-of-service attacks", Elsevier, 2009.
[4] D. Sisalem, J. Floroiu, J. Kuthan, U. Abend, H. Schulzrinne. "SIP Security", John Wiley and Sons, 2009.
[5] C. EY. "Detecting DoS attacks on SIP systems", 1st IEEE workshop on VoIP management and security, 2006.
[6] S. Ehlert, C. Wang. T. Magedanz, D. Sisalem. "Specification-based denial-of-service detection for SIP Voice-over-IP networks", Third international conference on internet monitoring andprotection, 2008.
[7] W. YS, S. Bagchi, S. Garg, N. Singh, T. Tsai. "SCIDIVE: a stateful andcross protocol intrusion detection architecture for Voice-over-IP environments", International conference on dependable systems and networks, 2004.
[8] A. Lahmadi, O. Festor. "SecSip: A Stateful Firewall for SIP-based Networks", 11th IFIP/IEEE International Symposium on Integrated Network Management, 2009.
[9] J. Fiedler, T. Kupka, S. Ehlert, T. Magedanz, D. Sisalem. "VoIP Defender: Highly scalable SIP-based security architecture", Proceeding of International Conferenceon Principles, Systems and Applications of IP Telecommunications, pp. 11–17, 2007.
[10] H. Zhang, Z. Gu, C. Liu, T. Jie. "Detecting VoIP-specific Denial-of-Service Using Change-Point Method", 11th International Conference on Advanced Communication Technology, pp. 1059-1064, 2009.
[11] D. Geneiatakis, N. Vrakas, C. Lambrinoudakis. "Utilizing bloom filters for detecting flooding attacks against SIP based services", Elsevier Journal of Computers & Security, pp. 578–591, 2009.
[12] M. Ali Akbar, Z. Tariq, M. Farooq, "A Comparative Study of Anomaly Detection Algorithms for Detection of SIP Flooding in IMS", 2nd International Conference on Internet Multimedia Services Architecture and Applications, pp. 1-6, 2008.
[13] A. Karim Ganame, J. Bourgeois, R. Bidou, F. Spies, "A Global Security Architecture for Intrusion Detection on Computer Networks", IEEE International Symposium on Parallel and Distributed Processing, pp. 1-8, 2007.
[14] R. Sekar et al. "Specification-based anomaly detection: a new approach for detecting network intrusions", in Proceedings of the 9th ACM conference on Computer and communications security, pp. 265-274, 2002.
[15] http://sipp.sourceforge.net/
[16] www.tcpdump.org
[17] Iran University of Science and Technology, Research Center of Information Technology, Network Research Group, SIP security page: http://nrg.iust.ac.ir/sip-security
[18] Z. Asgharian, H. Asgharian, A. Akbari, B. Raahemi, "A framework for SIP intrusion detection and response systems," International Symposium on Computer Networks and Distributed Systems, pp.100-105, 2011.
[19] Z. Asgharian, H. Asgharian, A. Akbari, B. Raahemi, "Detecting Denial of Service Attacks on SIP Based Services and Proposing Solutions." In Kabiri, P. (Ed.), Privacy, Intrusion Detection and Response: Technologies for Protecting Networks. (pp. 145-167). doi:10.4018/978-1-60960-836-1.ch006
[20] OPENSIPS, open source SIP proxy, http://www.opensips.org/
[21] M. Nassar, R. State, O. Festor, "Labeled VoIP data-set for intrusion detection evaluation", Proceedings of the 16th EUNICE/IFIP WG 6.6 conference on Networked services and applications: engineering, control and management, pp. 97-106, 2010.
[22] M. Nassar, R. State, O. Festor, "Monitoring SIP Traffic Using Support Vector Machines", Proceedings of the 11th international symposium on Recent Advances in Intrusion Detection, pp. 311-330, 2008.
[23] S. Ehlert, G. Zhang, D. Geneiatakis, G. Kambourakis, T. Dagiuklas, J. Markl, D. Sisalem, "Two Layer Denial of Service Prevention on SIP VoIP Infrastructures", Computer Communications, pp. 2443–2456, 2008.
[24] Angelos D. Keromytis, "Voice over IP Security", Springer, DOI 10.1007/978-1-4419-9866-8, 2011.
[25] G. Ormazabal, S. Nagpal, E. Yardeni, H. Schulzrinne, "Secure SIP: A Scalable Prevention Mechanism for DoS Attacks on SIP Based VoIP Systems", Proceedings of the 2nd International Conference on Principles, Systems and Applications of IP Telecommunications, pp. 107–132, 2008.