Received: 13 December 2016 | Revised: 20 June 2018 | Accepted: 17 August 2018

DOI: 10.1002/dac.3825

RESEARCH ARTICLE

WILEY

Anomaly-based DoS detection and prevention in SIP networks by modeling SIP normal traffic

Mahsa Hosseinpour | Mohammad Hossein Yaghmaee | Seyed Amin Hosseini Seno | Hossein Khosravi Roshkhari | Mohsen Asadi

Department of Computer Engineering, Ferdowsi University of Mashhad (FUM), Mashhad, Iran

Correspondence
Mohammad Hossein Yaghmaee, Department of Computer Engineering, Ferdowsi University of Mashhad, Mashhad, Iran.
Email: hyaghmae@ferdowsi.um.ac.ir

Summary

Due to the various features of Voice over Internet Protocol (VoIP), this technology has attracted the attention of many users in comparison with the traditional telephony system. However, with the growth of this technology, the security issues and protection of its users against different kinds of threats have been raised as an essential requirement. Session Initiation Protocol is a predominant protocol to initiate and terminate multimedia sessions in VoIP networks that provide simplicity and text-based features. Despite its mentioned advantages, these features impose several vulnerabilities on VoIP networks. Denial of Service attack, as one of the most common attacks against VoIP networks, is also a noted security issue in the Internet Protocol platforms. In such attacks, the attacker tries to prevent service from authorized users by consuming server resources. These attacks can be launched by sending out-of-sequence messages, malformed messages, and flooding different kinds of messages. In this study, a new anomaly-based method is presented for detection and prevention of these attacks. Normal traffic of a VoIP network is modeled by making a finite state machine, which is used for attack detection besides other proposed modules. Furthermore, a whitelist method is implemented using Bloom data structure for attack prevention. The proposed method is completely implemented and tested using different test scenarios. The obtained results show that by using proposed method, attacks can be detected more accurately with lower false ratios and delay in comparison with the existing methods.

KEYWORDS

VoIP network, SIP security, DoS attacks, finite state machine (FSM)

1 | INTRODUCTION

Voice over Internet Protocol (VoIP) technology allows users to establish telephone calls and other multimedia streams over the IP protocol. This technology offers numerous features for both users and providers such as low cost and flexible services that are difficult to realize in traditional Public Switched Telephone Networks (PSTN), which due to the closed networking environment dedicated to voice services, security threats are considered minimal. Unlike the PSTN, VoIP services are more vulnerable to security threats since they are based on the open Internet and may suffer from security threats like other Internet services.[1,2] As in PSTN, the main part of VoIP communications is a signaling protocol

responsible for establishing, modifying, and terminating sessions between users. Among different signaling protocols, the Session Initiation Protocol (SIP)[3] has been widely adopted for VoIP and multimedia systems. Since this protocol is text based and involves a simple connection process and easy implementation, it can be used as a target by the attackers to exploit its potential vulnerabilities. Therefore, to offer reliable services, security of this protocol should be provided. Different kinds of attacks have been identified against SIP-VoIP services up to now. One of these attacks, which is also one of the most alarming attacks on the Internet, is Denial of Service (DoS). DoS is perhaps the most powerful attack as it depletes the underlying resources so that legitimate users cannot have access to desired services.[4,5] The aim of these attacks is to make a service or application inaccessible by sending crafted messages or sending a huge amount of useless traffic.[2]

These kinds of attacks indeed violate availability as one of the security requirements. Since availability is one of the main features of the PSTN, ensuring the same level of this requirement is important to the widespread adoption of the VoIP services.[6] There are different kinds of DoS attacks against SIP-VoIP networks. In addition to DoS attacks against IP networks and transport layer, which are also valid for SIP networks, there are new kinds of attacks that directly target the SIP application layer itself. These attacks include SIP message flooding, SIP message payload tampering, and SIP flow tampering attacks.[2]

In this paper, we propose a new anomaly-based method to detect and prevent SIP-specific DoS attacks especially when the attackers target SIP proxy server. To this end, two different phases including training and testing are considered. We modeled SIP normal traffic in the training phase by constructing a finite state machine (FSM) from real calls of a VoIP network with no attack. Using the parameters extracted from the obtained FSM, three different kinds of attacks that lead to DoS in a SIP network are detected. Next, in the testing phase, considering the created FSM, flow of calls could be investigated. Also, three different kinds of flooding attacks could be detected using some fuzzy systems and a new equation called as Session Distance.[7,8] In each state of the FSM, a hierarchical tree structure is used to monitor the correctness of the message structure syntactically. Three different states are considered for attack detection. By placing the system in each of them, a prevention mechanism is applied, if it is required. Once a DoS attack is detected, a filtering-based mechanism using a whitelist is applied and so only messages from legitimate users can reach the server. To save required space for legitimate users' information, a Bloom filter data structure is used.

The rest of the paper is organized as follows: Section 2 includes an overview of the SIP protocol and different kinds of DoS attacks. The presented DoS detection and prevention methods are reviewed in Section 3. In Section 4, the proposed method including training and testing phases and prevention mechanism are presented. In Section 5, the implementation testbed and experimental results are discussed, and finally, Section 6 concludes the paper.

2 | BACKGROUND

2.1 | SIP overview

SIP[3] is an application layer protocol that is standardized by the Internet Engineering Task Force designed to establish multimedia sessions such as VoIP calls among the users. SIP operation is similar to the HTTP: it is text-based, has a request-response structure, and applies an authentication mechanism based on the HTTP Digest Authentication to authenticate its users. An SIP message consists of a Request-line that specifies its request or response type, followed by required header fields needed to describe each request or response and optionally the message body that can be found in some requests such as INVITE request.

Figure 1 shows a typical SIP message flow for call establishment between two user agents (UA). As can be seen, Alice (UAC) and Bob (UAS) are placed at two different domains. First, Alice's UAC sends an INVITE message to the proxy server to initiate a new call with Bob (#1). B's proxy server receives the INVITE and sends a 100 TRYING response back to the A's proxy server (#2) to indicate that it has received the INVITE and is processing the request. Since Bob is in another domain, by receiving the INVITE request, the proxy server performs DNS lookup to find IP address of the proxy server in Domain B and then it forwards the INVITE message to the B's proxy server (#3). Also, B's proxy server receives the INVITE message and sends back a 100 TRYING message to the A's server (#4). Using the location service, it finds current IP address of Bob and relays the message to the Bob's UAS (#5). Bob's phone receives the INVITE request, starts ringing, and sends back an 180 RINGING response (#6). When Bob answers the call, its UA also sends a 200 OK response (#9). These messages are returned to the Alice's UA by the servers (#7, #8, #10, and #11). To acknowledge the reception of 200 OK, Alice's UA sends an ACK message (#12). This message is sent directly from Alice's UA to Bob's UA. After this three-way handshake, the media session is independently established between two parties. Now, Alice

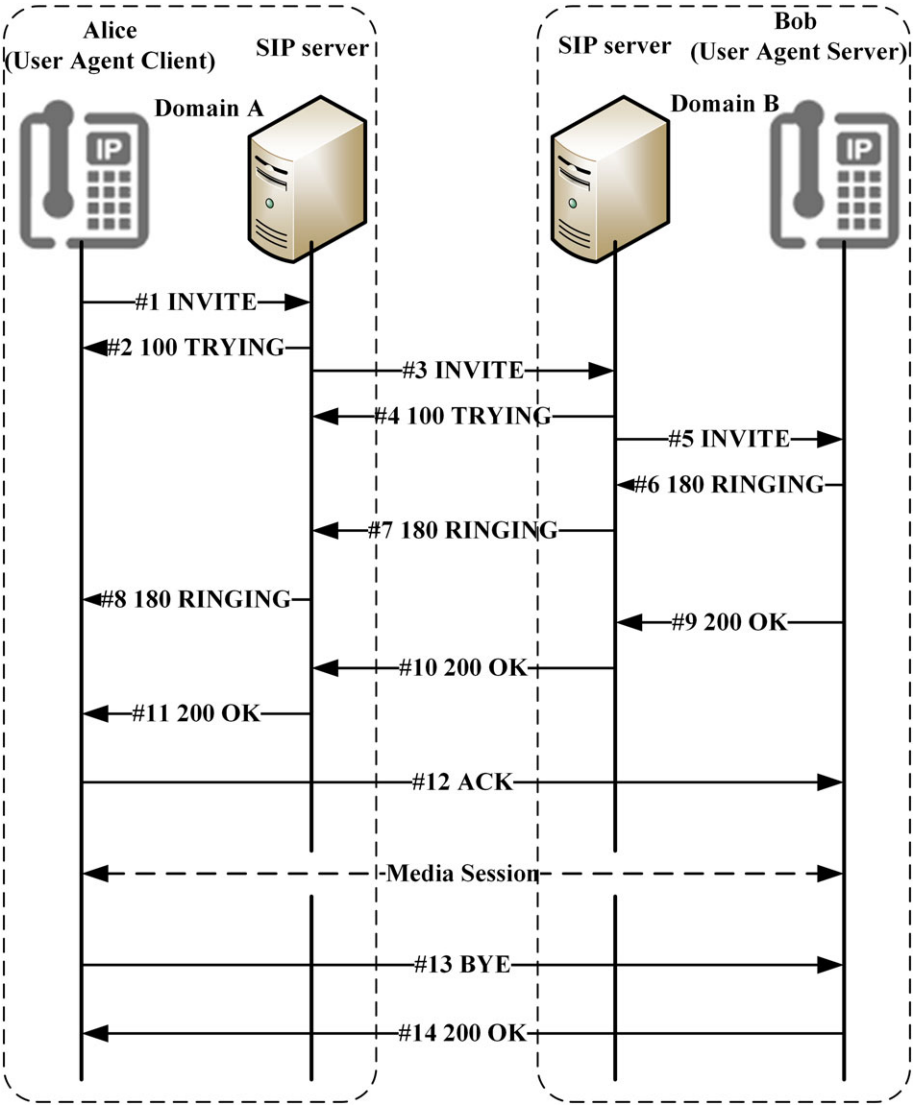

FIGURE 1 Message flow for establishing a connection in Session Initiation Protocol (SIP)

and Bob can send multimedia packets using the format to which they agreed. Finally, at the end of the call, one of the parties (here Alice) sends a "BYE" request (#13) to Bob and he responds with a "200 OK" (#14), and the call is terminated.

2.2 | DoS attacks in SIP

Achieving a security level, equal to that of traditional Telephone system, is one of the important challenges in VoIP systems. One of the threatening attacks against SIP services is DoS that renders the target component useless by consuming its resources with the aim of denying the legitimate user's access to the network services. Several types of DoS attacks can be launched against SIP services that target signaling protocols, media protocols, and also physical components. In more complex situations, the attackers exploit tools to attack the target from different sources in a distributed manner. In such a situation, which is known as Distributed Denial of Service (DDoS) attack, attack detection and prevention are more complicated. The target of a DoS or DDoS attack can be SIP UAs or proxy servers in a VoIP network.[2] When the proxy server does not work properly, the number of users without access to their desired services rises. Thus, the attackers target the proxy server to increase the impact of the attack. CPU, memory, and bandwidth are three required resources for SIP server proper functioning that are targeted in DoS attacks.

2.2.1 | CPU

When the proxy server receives each new message, it should parse the message, process it for some purposes like authentication, fulfill transaction mapping, and finally forward the received message. In this case, server's limited CPU resources are wasted by a large number of malicious requests. One of the important vulnerabilities of SIP server is the computational complexity of the authentication mechanism (it consumes 80% of the CPU usage of the proxy server[9]). This process is used by attackers to send numerous messages needing authentication to exhaust SIP server CPU resources. In addition, there are operations that need other components' output such as database lookup or DNS service, so CPU must wait until completion of their execution.

2.2.2 | Memory

By receiving a new SIP message, the SIP server needs to store message information in its buffers for processing purposes. The required time and amount of space for keeping these data depend on the server mode: either stateful or stateless. By creating each new transaction, its information is stored in the memory until it is closed or timed out.[10,11] SIP protocol is similar to Transport Control Protocol (TCP) in some ways as it brings vulnerability to analogous attacks. TCP needs a three-way handshake for establishing a new connection. A malicious user can exploit a vulnerability in this process to cause a DoS attack by consuming server's memory. One of the most common types of these attacks is TCP-SYN flooding attack. The attacker floods the TCP server through the requests with the SYN flag set. For any new request, the TCP server allocates appropriate space to handle it and sends back a TCP-SYN/ACK message. Since the received TCP-SYN messages have a spoofed IP address, the corresponding SYN-ACK responses do not refer to a real system. The attacker keeps on sending new TCP-SYN requests, and the server reserves resources to serve them until the related TCP timer expires and the allocated spaces are released.[7,10] Since call establishment using SIP is based on a three-way handshake, SIP servers are also vulnerable to these attacks.

2.2.3 | Bandwidth

In this case, the victim is flooded with a number of messages that are more than the handling capacity of the network. These attacks waste the capacity of the physical link connecting SIP server to the network and, as a result, the server cannot practically provide any services. Due to the link occupation with malicious packets, regular packets are discarded before reaching the server. This problem is common in networks and is not specific to SIP.[10]

In general, DoS attacks against SIP-based VoIP networks are divided into three categories: (1) SIP message flow tampering. (2) SIP message payload tampering or sending malformed messages, and (3) SIP message flooding.[2] In this paper, these attack groups are applied to focus on attacks detection and prevention.

2.2.4 | SIP flow tampering

In a real-time communication such as SIP-based communications, the call participants make a connection with each other and exchange desired content constantly. A malicious user can target these ongoing connections in order to disturb users' calls through manipulated messages. To reach this goal, the attacker uses different bogus signaling messages. These messages are the same as legitimate ones, except they are sent from an attacker by a spoofed address. BYE, CANCEL, UPDATE, Re-INVITE, and REGISTER requests can be applied to launch such attacks.[12]

Moreover, this category contains, the efforts made by attackers to disturb sequence of messages. In this case, the attacker can put system's accessibility at risk by creating numerous pending requests especially when the proxy server is stateful. When parser tries to match the requests and responses to the corresponding sessions, such behavior can lead to a long communication delay. As an example, the attacker can send ACK messages before reception of 200 OK responses.

2.2.5 | Malformed messages

Given that SIP is a text-based protocol, it provides a good opportunity for malicious users to alter different fields of a well-formed message. In this type of attack, the attacker tries to manipulate payload of a SIP message by inserting incorrect content to it or leaving a field without any value. Such attacks lead to a delay in parsing the message or decision-

making processes that affect system's accessibility and cause DoS attack. These attacks can also be applied to uncover vulnerabilities of the target. In this category, unintentional attacks may occur due to the poor implementation.[13]

2.2.6 | SIP message flooding

One of the easiest and most common ways to perform DoS attack is to flood network's components with numerous SIP messages. These attacks can have different causes: (1) planned attacks designed to damage the VoIP network and (2) unintentional attacks due to the bad configuration and wrong implementation of SIP devices. Detection and prevention of both attacks are necessary since the obtained results are similar. Different kinds of SIP messages can be used to cause a flooding attack. Among them we can name REGISTER flooding attacks, INVITE flooding attacks, BYE flooding attacks, and Ping flooding attacks (using OPTIONS messages).[14]

3 | RELATED WORKS

Several security solutions are proposed to detect and prevent SIP-VoIP related DoS attacks. Two main strategies are used to detect these attacks: (1) pattern (or signature)-based detection methods and (2) anomaly-based detection methods. In signature-based methods, premade attack patterns are considered as rules. If the current traffic matches the stored pattern, an attack can be detected. These methods are only able to detect attacks that have already occurred. To detect a new attack, like antiviruses, their pattern's databases should be updated with signatures of the new attack. In anomaly-based methods, normal traffic behavior is used to detect attacks. It means that some rules are derived from normal traffic behavior. Then, if the current traffic behavior is not consistent with specified rules, the abnormal behavior or attack can be detected. Unlike the previous method, in anomaly-based methods, new attack detection is possible. In this paper, we review different proposed methods that cover both abovementioned methods.

An intrusion detection architecture, named SCIDIVE, is presented to detect some attacks such as DoS attacks induced by the malformed messages and also media stream-based attacks in a stateful and cross protocol manner.[15] In Rezac et al,[16] a penetration test simulator is implemented to carry out several attacks on SIP servers through different applications and algorithms. The application sends test results about the potential system risks to testers. Moreover, it proposes recommendations to them to ensure high level security against threats. In Ehlert et al,[11] a two-layer architecture is presented for detecting DoS attacks. Flooding SIP messages, malformed messages, and DNS-related attacks can be detected using the proposed solution. In the first layer, flooding attacks are detectable by adding threshold values into the firewall. To detect flooding attacks, Hussain and Nait-Abdesselam[17] introduced a new parameter named Critical Number to inform the proxy server about the number of each kind of messages that a UA can handle. A stateful language named VeTo is presented in Lahmadi and Festor[18] to describe SIP vulnerabilities and corresponding countermeasures. The rules of this language are extracted automatically and executed with the use of Secure SIP engine.[19] Using SIP specification and its transaction's state machines and also a rate controller filter, Secure SIP can detect a sequence of messages and flooding attacks, respectively. In Li et al,[20] an anomaly-based framework is proposed for detecting malformed messages. This method presents a common pattern for all messages that specifies message format and a specific pattern for each kind of messages.

Regarding the importance of self-healing and as a result self-diagnosing in heterogeneous networks, in Lu et al,[21] a distributed self-diagnosis algorithm using casual graph modeling is proposed. The authors implement their algorithm in an IMS platform and effectively could identify the primary cause through tests' executions. To overcome the drawbacks of using VoIP services in distributed wireless networks, De Rango et al[22] proposed a novel metric based on the E-Model to increase the number of admitted calls in addition to improving QoS. Due to the sudden changes in network traffic during flooding attack, threshold-based methods were presented as common detection methods. In order to compute threshold value, in Reynolds and Ghosal,[23] cumulative sums are proposed. This value is statistically obtained by calculating the number of INVITE requests and corresponding 200 OK messages. In Sengar et al,[24] an anomaly-based method, named vIDS, is presented to detect different attacks against UAs by the help of protocol's state machines and their interactions. In Rebahi et al [25] a method based on change-point detection using CUSUM algorithm is proposed for detecting flooding attack. Unlike the presented method in Reynolds and Ghosal,[23] they only used INVITE messages. In Li et al,[26] the proportion of different messages is applied for calculating CUSUM statistical value in different time periods by the help of sliding windows. In Sengar et al,[27] the use of statistical methods is proposed for detecting flooding attacks. The authors calculated the probability distribution of messages in training and test phases. Next, they compared

the obtained results using Hellinger Distance (HD) to obtain their similarities by calculating probability distribution distances. An alarm is generated if the measured distance is more than the defined threshold value.

In Tang et al,[28] a three-dimensional statistical sketch data structure and HD method are designed to detect and prevent flooding attacks. The sketch data structure lets the authors have an independent probability distribution for each kind of SIP messages. In Zhang et al,[29] an entropy-based method, named advanced entropy-based, is proposed for detecting low-rate DDoS attacks. The attack can be detected by comparing traffic entropy with normal traffic entropy as a reference. Tang and Cheng et al[30] showed that CUSUM and HD-based detection schemes are not effective at low rate flooding attacks. Thus, they proposed a discrete wavelet transform–based detection method of the SIP stealthy flooding attacks. In Asgharian et al,[31] by incorporating specification and anomaly-based methods, an intrusion detection system is proposed. This system uses a state machine and threshold values to be able to detect attacks.

In Geneiatakis et al,[7] Bloom data structure is used to detect flooding attacks. Flooding attack is considered as the result of the sending INVITE messages from one or several sources and also TCP-SYN like attacks in this study. For attack detection, Session Distance concept, which shows the relation between INVITE, 200 OK, and ACK messages, is defined. Also in Roh et al,[32] a detection method using Bloom data structure is proposed. For this reason, a whitelist is created using legitimate users information, and the possible attack is detected using a nonmembership ratio. To prevent flooding attacks, in Chen and Itoh,[5] a whitelist-based method using filtering according to the call history is proposed. If the attacker sniffs legitimate user information, this method is not able to prevent such attacks. To detect DoS attacks against UAs, the authors of Seo et al[8] introduced a tree-based data structure named SIPAD. This method uses defined transaction state machines in SIP standard and can only detect different DoS attacks during transactions. In previous studies,[33,34] the authors used log files and audit trails to detect improper use of VoIP network. The privacy and huge size of users' information encouraged the authors to apply hash functions for anonymity purposes. Also, an entropy-based method is used to compare the information and detect the attacks. In Marchal et al,[35] a particular telephony DoS attack named mimicry TDoS is studied. The authors elaborated malformed messages that have a slight difference with a syntactically correct one. They modeled the system using optimization methods and classifiers to classify messages more precisely. In Safarik et al,[36] attack detection by a monitoring mechanism through distributed monitoring nodes, which the main part of them is honeypot application, is proposed. To analyze the gathered information deeply, a neural network is designed and implemented for automatic classification and therefore detecting some type of attacks. In Golait and Hubballi,[37] the normal traffic profile is created by the help of a probability distribution. Considering the obtained distribution in different time slots, threshold values are defined and used for attack detection. In Shah and Dave,[38] an anomaly-based method is presented to detect malformed messages. This method prevents malicious messages using a whitelist including IP address of legitimate users. In Cadet and Fokum,[39] an IPS is presented by changing rules of Snort. Since it uses threshold values for flooding attacks, it can only detect flooding from one source; hence, the distributed and low rate flooding attacks are not detected. In Tas et al,[40] a solution is presented for detecting and preventing DDoS attacks induced by using UDP as the transport protocol, through collecting packet information to set threshold values dynamically. To detect and prevent INVITE flooding attacks with a preventive method, two UDP messages (WAIT and GO) are used in Raza and Raza[41] to inform the caller if the proxy is busy. After simulating a VoIP system in Ghafarian et al[42] and making flooding attacks against it, a Snort-based IDS system is designed for attack detection and prevention.

4 | THE PROPOSED METHOD

In the present study, we propose an anomaly-based system to detect and prevent DoS attacks in SIP-based VoIP networks. To this aim, we consider out-of-sequence messages, malformed messages, and flooding messages attacks that lead to DoS in these networks. In addition to detecting a huge amount of sent messages by the attacker, in the case of flooding attacks, the low-rate and distributed flooding attacks are considered to detect. Because of the important role of the proxy servers, detecting DoS attacks against them is intended. We try to improve accuracy of the system by detecting more attacks with fewer false alarms rate, which imposes slight delay to the VoIP system. We consider two training and testing phases to detect and prevent DoS attacks. To extract information of calls from normal traffic of the network in an anomaly-based manner, we create a FSM in training phase. Next, in the testing phase, we detect the mentioned attacks using different solutions. At the end of each time window, if any attack is detected, the system will be placed in the Attack state, until the end of the current time window. After attack detection, a whitelist method is used by the help of Bloom filter data structure to prevent attack packets from reaching the server.

4.1 | Training phase

To obtain the parameters of calls and behavior of a real VoIP network traffic to treat it as a normal traffic for attack detection, we model VoIP traffic by creating an FSM. In addition to the network design, planning, and management, one of the applications of traffic modeling is to identify and characterize traffic for purposes such as security. To communicate between different parts of computer networks, various protocols have already been created that specify rules and format of messages needed for data exchange. Such protocols have been standardized and presented in the form of specifications by organizations like Internet Engineering Task Force. These specifications play an important role in different fields such as implementing effective security methods.[43-45] SIP protocol (RFC-3261) is one of these specifications, which its direct use for assessing the behavior of the protocol in a real network is not appropriate. The advantages of creating FSM can be summarized as follows:

- Producing a protocol specification readable by machines is a time-consuming and error-prone task.
- Protocols are constantly evolving with new functionalities and message formats that render the previously defined specifications incomplete or deprecated.
- This approach also has the advantage of capturing unpublished or undocumented features automatically, thus obtaining a more complete and realistic specification of the implemented protocol.

Accordingly, in this phase, calls of a VoIP provider at different times of the day are applied for making FSM. To extract required call parameters and construct FSM, a parser module is created. To observe different call sequences and their exchanged messages, it is required to keep session's information. Using this module, important parameters such as FROM, TO, and Call-ID fields in each message of the existing sessions, including requests and responses, are extracted. Each SIP session could be determined by the <From-tag, Call-ID> two tuples. Considering the current state of the session and type of the received message, if there is a state to change, the current state of the call session is updated along with some information about this state changing. Otherwise, a new state is created and its information is stored in a table of the database. By finishing each session, corresponding stored information is deleted from related database tables to release allocated spaces. Therefore, each state in the FSM represents a SIP message (request or response) and transition between states, means inputs of FSM, show the message that leads to that transition and represents the flow of calls. Because of the privacy considerations, hash of these values is held.

In addition to session information and sequence of exchanged messages, the sum of observed messages in each state transition and the average of time differences between any two states are calculated. These two parameters are used by fuzzy systems to detect flooding attacks since the value of these parameters will change during attacks. It should be noted that behavior of telephony traffic is different on various days of a week and various time durations of a day.[46] For taking into account these differences, we calculated the mentioned values at different time intervals.

4.2 | Testing Phase

Generally, three different attack states are assumed in the proposed system, which the system will be active in one of them in each step considering different conditions. The states are known as Normal, Alarm, and Attack states. At first, the system is placed in Normal state. At this state, calls are normally established and all the messages reach the server. Also, at this state, information of the normal users that terminates their calls successfully is registered in a whitelist. By detecting each kind of attacks, the system is placed at Attack state. At this state, the system uses whitelist information to send the messages from normal users to the server. In addition to these two states, which are common between different attacks, an Alarm state can be used by fuzzy systems in order to detect flooding attacks (from one source or distributed). However, we give it up for future works. The flowchart of the entire proposed DoS detection algorithm is shown in Figure 2.

Because the number of calls at different durations of a day changes, like the training phase, time intervals are considered. Therefore, the values extracted in each time interval are compared with those at the same time interval in the training phase. Moreover, to detect attacks in shorter times, smaller time windows are assumed. In addition to preserving the flow of calls in each time window by observing a specific message, the number of observed messages in each state transition and average of time differences between each two states are calculated.

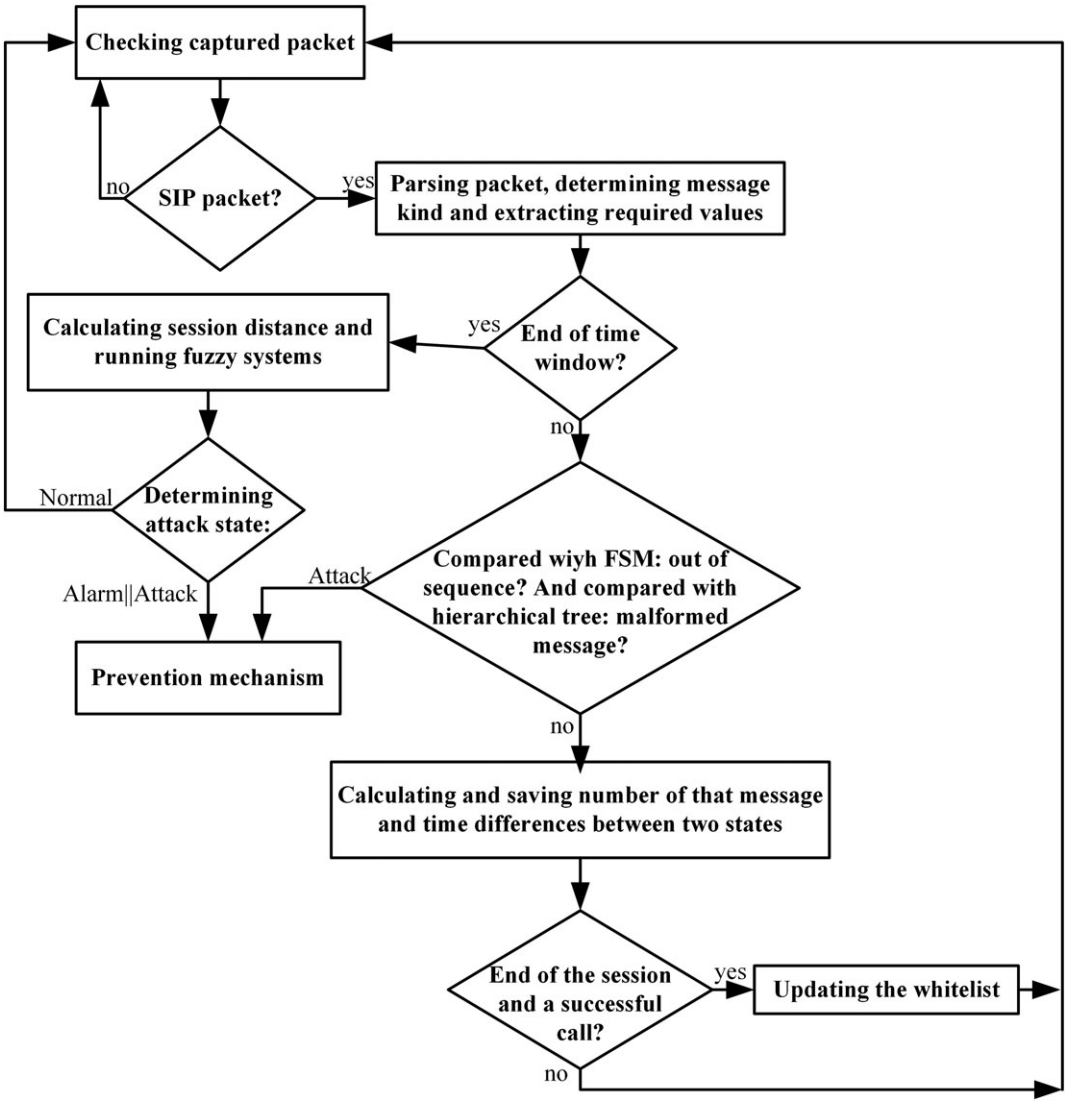

FIGURE 2 Flowchart of the proposed Denial of Service detection system

4.2.1 | SIP message flow tampering detection

As mentioned earlier, in the proposed method, flow of calls is compared with those observed in the created FSM. By the help of created FSM in training phase, we can detect most of the out-of-sequence messages. It is possible that an out-of-sequence message, which is sent from an attacker, places at an allowed state of the FSM. In such cases, detecting an attack is impossible in this step. The sequence of messages is also effective in a more quick detection of flooding attacks, especially when the attacker floods the system using a combination of SIP signaling packets (with or without legitimate users' information sniffing). Such messages could be considered as malicious ones if they are not observed in allowed states.

4.2.2 | SIP message flooding detection

In the proposed method, flooding attack due to the huge amount of SIP messages (from one source or in a distributed manner) and session flooding attacks (low-rate and TCP/SYN-like flooding attacks) are considered. To detect such attacks, we use fuzzy systems and a relation called Session Distance. In addition to the considered time intervals, we determined time windows. At the end of each time window, based on the observed SIP messages, some fuzzy systems are executed.

Detecting normal and distributed flooding attacks

We use some fuzzy systems to detect these kinds of attacks. This part is designed as a way that system administrator can specify system's sensitivity to these attacks. Therefore, we define 10 sensitivity degrees so that by increasing selected degree, the sensitivity of the system will increase. In this paper, Mamdani approach is used as fuzzy derivation method. As previously mentioned, there are some fuzzy systems in the proposed algorithm that each one has two inputs and one output variables. Inputs of each fuzzy system are the number of messages in each transition and average of time differences between each two states, while its output is attack severity. The correct choice of membership functions has a significant impact on system's performance. In applied fuzzy systems, the membership functions are determined through practical tests and trial and error. For example, in Figures 3–5, membership functions for inputs and output of INVITE to 401 (Un-Authorized) messages are shown.

After specifying the membership functions, fuzzy rule base must be designed. The fuzzy rule base is similar to the fuzzy inference engine core. It is composed of some If-Then rules for decision-making. Note that if rules base is extensive, then the fuzzy system is complicated and its speed is degraded. In Table 1, the applied rules in the proposed method are presented. The intended logic to make these rules can be stated as follows: the attack severity increases by reducing or excessive increasing time averages and also increasing the number of messages.

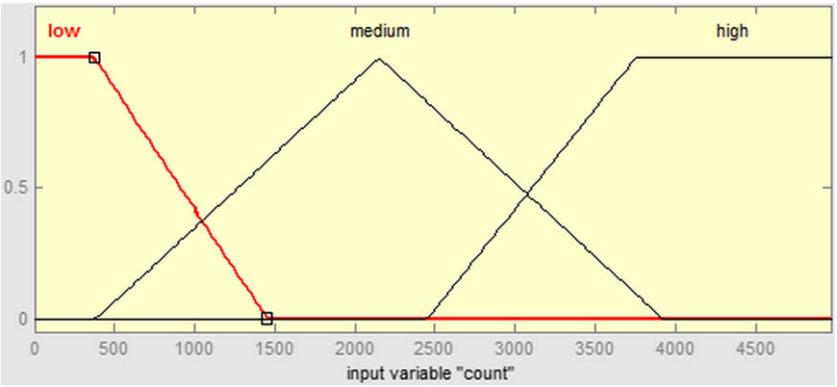

FIGURE 3 The membership function of input variable "count" (number of messages)

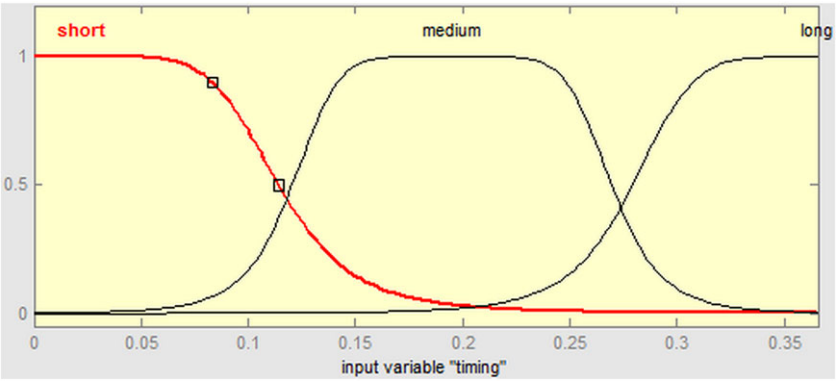

FIGURE 4 The membership function of input variable "timing" (average of the time difference between states)

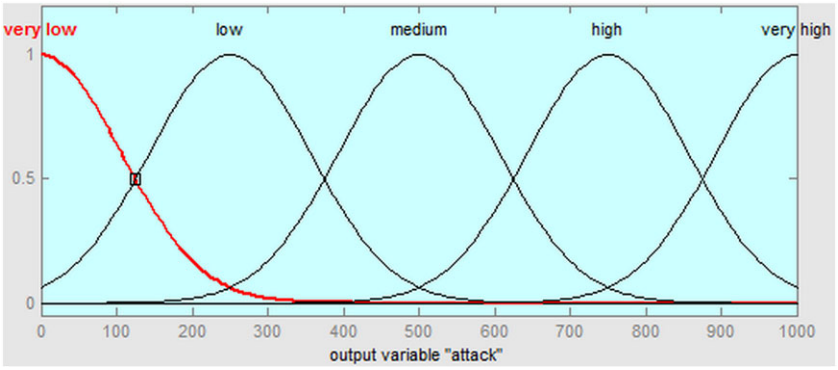

FIGURE 5 The membership function of output variable "attack severity"

TABLE 1 Fuzzy rule base

Timing Count	Short	Medium	Long
Low	Low	Very low	Low
Medium	High	Medium	High
High	Very high	High	Very high

As previously mentioned, in the proposed method, the sensitivity of the system to flooding attack, could be determined by the system's administrator. Accordingly, fuzzy rules are assumed as weighted rules with administrator's included sensitivity degrees. The expert's opinion, practical tests, and trial and errors are used to achieve appropriate values for weights. When system's sensitivity degree is set to 5, the fuzzy rules along with considered weights are as follows:

1. If (count is low) and (timing is short), then (attack_severity is low) (0.5).
2. If (count is low) and (timing is medium), then (attack_severity is very low) (0.6).
3. If (count is low) and (timing is long), then (attack_severity is low) (0.5).
4. If (count is medium) and (timing is short), then (attack_severity is high) (0.4).
5. If (count is medium) and (timing is medium), then (attack_severity is medium) (0.1).
6. If (count is medium) and (timing is long), then (attack_severity is high) (0.4).
7. If (count is high) and (timing is short), then (attack_severity is very high) (0.3).
8. If (count is high) and (timing is medium), then (attack_severity is high) (0.4).
9. If (count is high) and (timing is long), then (attack_severity is very high) (0.3).

To evaluate rules, first of all, inputs are fuzzified and then applied to the rules' premier section. In this system, AND fuzzy operator is used to extract a number representing the assessment of the rules' premier section. Next, the derived number is applied to the inferior section. Also, union operator is used to merge the results of applying fuzzy rules. Consequently, a central average de-fuzzier operator is applied to derive a real output. The experts' opinion and trial and errors are used to define thresholds for determining system's attack state. Three numeric ranges are considered for Normal, Alarm, and Attack states. A sample of proposed fuzzy system execution, for input values with count = 2485 and timing = 0.1828 (seconds), is depicted in Figure 6. The obtained output shows a Normal state for this special state transition. If the attack situation is observed at the end of each time window only for one state transition, the system will place at Attack state for the next time window.

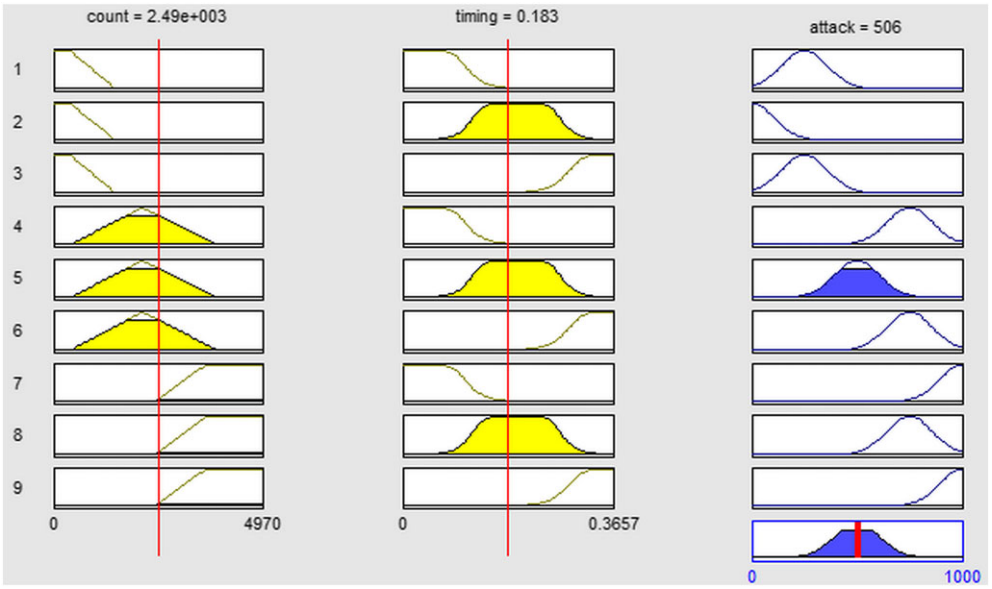

FIGURE 6 System's outputs for input values (2485,0..1828)

Detecting session flooding

When the attacker tends to flood the server with numerous open sessions and also low rate attacks, a number of open sessions must be controlled. A new session will initiate between two call participants by sending an INVITE request by the caller and will complete by sending ACK message to acknowledge receiving callee's 200 OK responses. In normal conditions, the number of INVITE messages and the number of corresponding ACK messages at initiating sessions are approximately equal. Based on this fact, an equation for controlling this proportion, called Session Distance, is presented in (1):

$$\text{session distance} = \frac{\#\text{INVITE messages}}{\#\text{ACK messages}}. \tag{1}$$

Since SIP uses UDP as transport protocol by default, resending INVITE messages to cope with packet loss in UDP is considered in its standard. Considering these details, this proportion in normal conditions will be almost 1 or slightly more. When the attacker attempts to send INVITE messages with a low rate and increase it slowly, by increasing the number of server sessions and degrading its performance over the time, it will not be able to response by sending 200 OK messages, and accordingly, no ACK message is sent by the caller. Hence, the proportion of these two messages will increase gradually. In TCP/SYN-like attacks, which the attacker uses spoofed INVITE messages, when the server sends back corresponding 200 OK responses, no ACK message will be sent as there is no real UAC in order to acknowledge the reception of 200 OK message. Furthermore, this approximate proportion is not held and its gradual increase would be noticed. In both cases, for an accurate attack detection, we use a threshold value calculated using numerous tests. It should be noted that defined timer for receiving ACK messages in RFC-3261 is included in Equation (1). This timer is equal to $64 \times T_1$, where T_1 is 500 ms by default. If the desired ACK is not received in defined time period, that transaction will be terminated, and its allocated space in the server will be released. Therefore, by taking into account this timer, we can delete that INVITE request from the numerator.

4.2.3 | SIP message payload tampering (malformed messages)

In the proposed method, we use provided syntax in RFC-3261 to detect malformed messages. The correct format and syntax of SIP messages are presented in Augmented Backus-Naur Form (ABNF) format that also provides the correct structure of messages. We apply the developed form of the hierarchical tree structure presented in Seo et al.[8] In the proposed method, in each state of the created FSM, only some SIP messages and some header fields are allowed for each SIP message (based on the RFC-3261). Each header field can only have some defined rules. Consequently, using this hierarchical structure, first we can check the mandatory, allowed or not allowed, and optional parts of each message to prevent the attacker inserting or removing any field to attain poor network operation. Second, each field syntactically is checked with the help of rules and sub-rules of the SIP standard. For this purpose, we convert the existing ABNF format to the regular expression format for comparison purposes. A part of this tree-based hierarchical structure is shown in Figure 7. Mandatory, optional, and not-allowed header fields are introduced in RFC-3261 for each request and response. In Figure 7, mandatory and optional fields are illustrated by solid lines and dashed lines, respectively. Also,

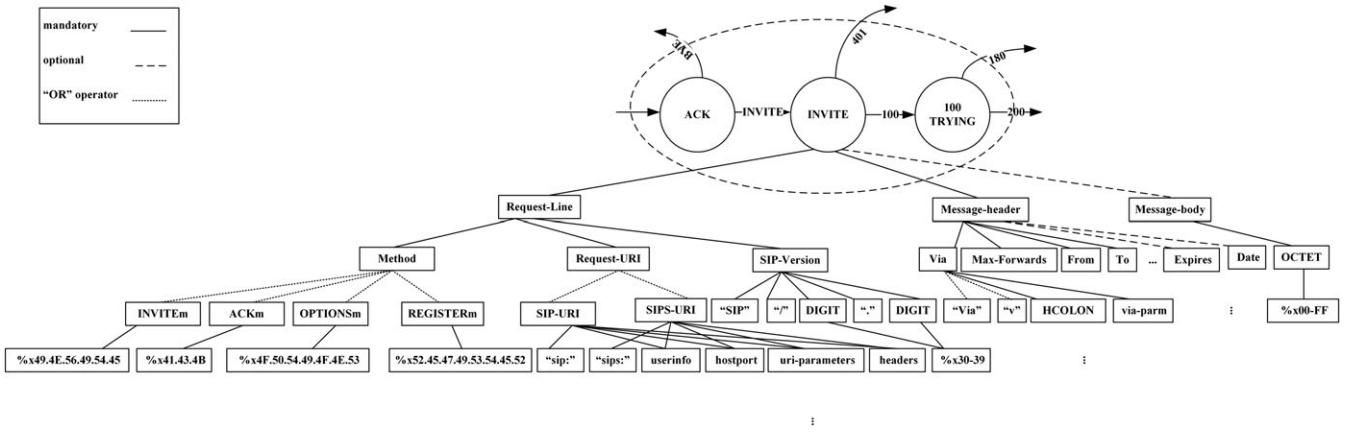

FIGURE 7 Tree-based hierarchical structure for detecting malformed messages

"OR" operator, when only one of the existing fields is sufficient, is depicted by dotted lines. Therefore, by receiving a SIP message in each state of FSM, the presence or absence of each header fields can be investigated. Then, the syntax of the mandatory and optional fields (if there are any) is compared with RFC's rules.

4.3 | Attack prevention mechanism

As mentioned before, we use the whitelist-based mechanism to prevent attack packets reaching the server. To reduce required space for legitimate users' information in the whitelist and also the complexity of comparisons, we apply statistical Bloom filter data structure for whitelist implementation. A Bloom filter is a space-efficient probabilistic data structure that is used to test whether an element is a member of a set in a constant delay.[47] Using a Bloom filter, a set $A = \{a_1, a_2, ..., a_n\}A = \{a_1, a_2, ..., a_n\}$ with N elements is described by vector V with m bits initially set to 0. Here, there must be k different hash functions defined. By applying all k hash functions to each element of set A, k random numbers in the range of *1 ... m* are generated and corresponding bits are set to 1 in vector V. To investigate an element whether belongs to the set A, all k hash functions are applied to that element. If all the corresponding bit positions are 1, the element belongs to set A; otherwise, it is not a member of that set. However, there is a possibility that this conclusion be incorrect; when the intended element is a member of the vector, the result certainly is true, but it is possible that input value is not a member of the set, but it is announced the member of the set incorrectly (false positive). In order to create a whitelist of users that call each other successfully, we apply caller IP address, URI address in From header field, and URI address in To header field; ie, <$caller_{IP}$,$FROM_{URI}$,TO_{URI}> triplets.[32] The procedure followed based on this Bloom filter is shown in Figure 8. To design this filter, it is required to determine the number of hash functions (k), the size of vector V (m), and the number of input elements (N). For conducting our experiments, we also consider 20 000 inputs. Values of k and m should be determined in a way that the obtained false positive value be acceptable. For

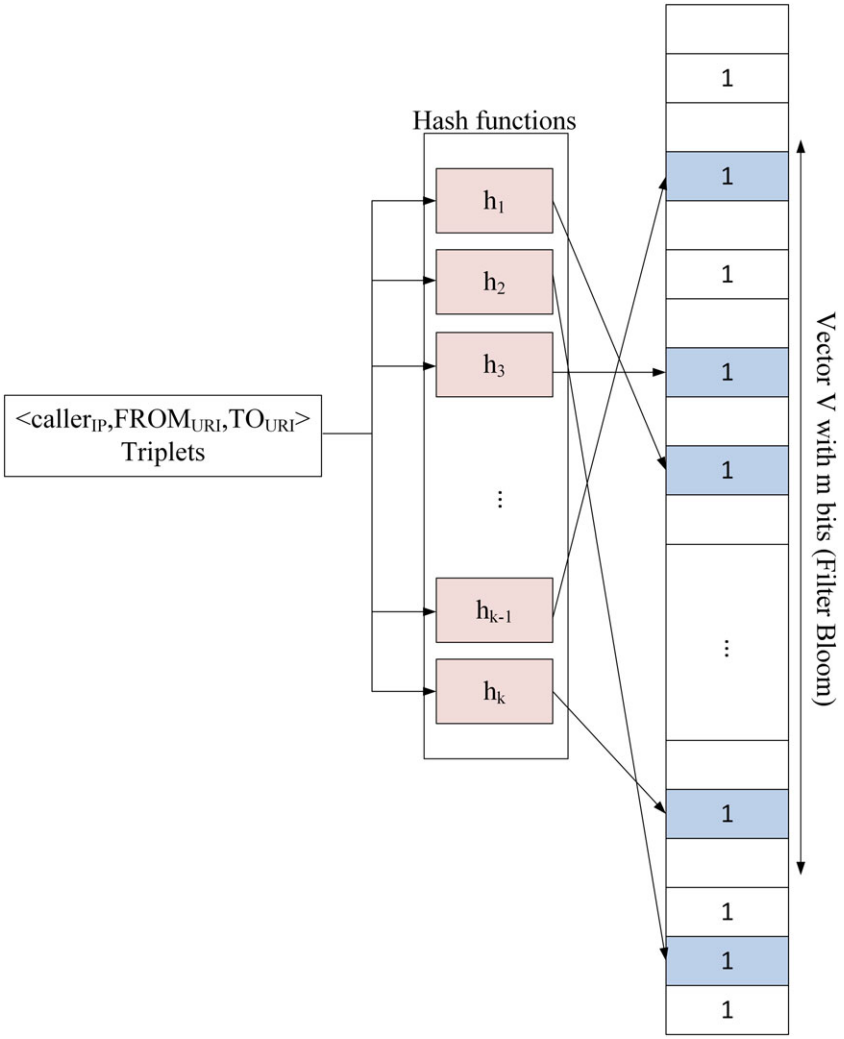

FIGURE 8 The Bloom filter method

this purpose, we apply relation (2) to calculate false positive probability when filter's parameters, means k, m, and N, are specified.[47]

$$p_{\text{err}} \approx \left(1-e^{-kN/m}\right)^{k}. \tag{2}$$

By assuming the maximum acceptable false positive probability and number of input elements, as they are respectively 0.1% and 20 000 in Roh et al,[32] we consider the filter size (m) and the number of hash functions (k) equal to 345 700 and 5, respectively.

In the Attack state, messages from users that do not belong to the whitelist are prevented, whereas it is possible that they are from legitimate users who make a call for the first time. In such circumstances, it should be noted that if there is not such prevention mechanism, the server can serve to none of the users. Also, using this method, even in attack conditions, the server can serve to the legitimate users that have already made a call successfully. Furthermore, information of more legitimate users will be kept in whitelist over the time and consequently fewer calls from legitimate users will be prevented.

5 | IMPLEMENTATION RESULTS

The proposed DoS detection and prevention system is fully implemented and is compared with similar methods.

5.1 | The test bed

In the proposed method, in addition to the traffic of a VoIP network for training phase, a SIP proxy server and some test tools are needed to generate normal and attack traffics in the testing phase. Figure 9 shows the implemented test bed for testing the system.

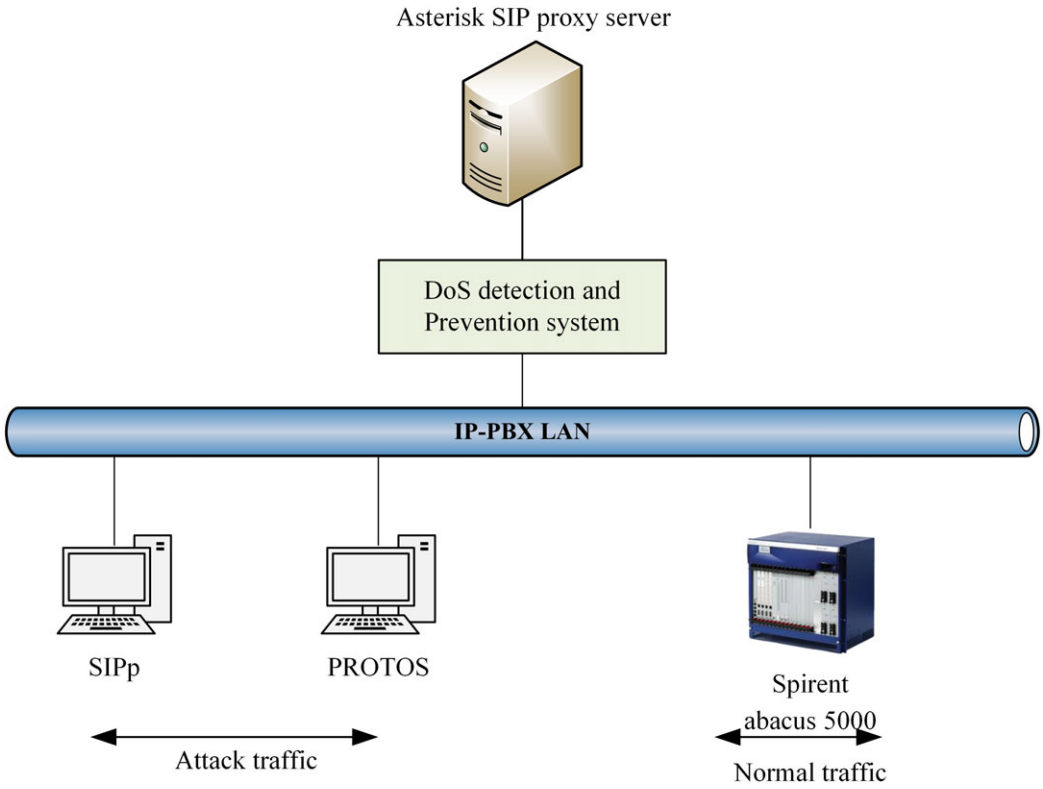

FIGURE 9 The implemented system test bed

5.1.1 | SIP proxy server

We use Asterisk proxy server as a service provider for SIP calls. Asterisk is the most popular open source VoIP telephone system in the world, which is the base for many available IPPBX. Asterisk is based on C programming language and could be installed on different operating systems such as Linux NetBSD, UNIX, Solaris, and Mac OSX. Furthermore, there are some versions of this software that can be executed in Windows operating system. Even though Asterisk services could be operated using common computers and servers with their possible computing power, the popularity of Asterisk and diversity of its services have made producers to use a combination of Linux and Asterisk platforms to produce communication equipment. This software uses UDP and TCP transmission protocols to receive and send SIP messages.

5.1.2 | SIP normal traffic generating tool

We use Spirent Abacus 5000 device to generate SIP normal traffic. This device can generate traffic with different transmission rates and distributions. The device is applied to various tests including interoperability, performance, scalability, and quality of audio and video over IP networks. Using this device, hundreds or even thousands of defined UAs could call each other in a normal way, with Spirent playing the role of each UA.

5.1.3 | Attack traffic generation tools

The required traffic for flooding VoIP system and making malformed messages is created using SIPp* and PROTOS† tools, respectively. We use the presented VoIP test package in Kali Linux to achieve these tools. SIPp is a free open source test tool for measuring SIP performance and could generate thousands of SIP calls in the system. Using this tool, it is possible to define different SIP traffic generation scenarios with different rates by using a certain type of SIP signaling packet or combination of them. In addition to the default embedded scenarios, we can generate our arbitrary scenarios through the XML files as external scenarios. With this capability, it is possible to define attacks using different SIP messages. Furthermore, this tool dynamically shows the results and statistics of in-process tests.

A set of test tools (PROTOS Test-suite:c07-sip) in the PROTOS test tool present 4527 malformed SIP messages and also permit to insert new ones. This tool is used to assess the security level and robustness of SIP implementations. This test set makes syntactical errors in INVITE messages. There are a number of test groups, each including a number of test cases and each test case, in turn, contains an exceptional element. An exceptional element is designed to provoke an incorrect behavior in the target. Among 4527 test cases, 2426 of them correspond INVITE header fields and the others belong to SDP protocol, which is ignored in our tests. In addition, some secure rules were added to this test set in Seo et al[8] that are also used in the present study for comparison purposes.

5.2 | Experimental results

We created the required FSM using calls of a VoIP provider. A part of such FSM, which was created using 5 minutes of VoIP network calls, is shown in Figure 10. These calls belong to the time period from 9 to 10 AM (9:16′-9:21′). In this figure, each state indicates a message type and each edge shows the message that leads to a state transition. The edge values include the message that leads to state transition (input) and the number of such messages (output). As mentioned earlier, also the average time differences in each state transition are extracted, but they are not shown in this figure for more readability and clarity.

To evaluate scalability and space complexity of the obtained FSM over the time, we measured its required space considering VoIP provider calls in different time spans. The obtained results for 2 hours of telephony calls are depicted in Figure 11. The results show that approximately, after 1 hour, almost all the possible FSM states are observed. As a result, the size of FSM is not changed and is remained at about 7 Kbit.

The size of time windows has a great impact on the attack detection speed and accuracy. We used false ratios to determine the best size for time windows. In Figure 12, the calculated false positive and false negative ratios considering different time windows are shown. Since flooding detection is done at the end of each time window in our system, we

*http://sipp.sourceforge.net/.

†https://www.ee.oulu.fi/research/ouspg/PROTOS_Test-Suite_c07-sip

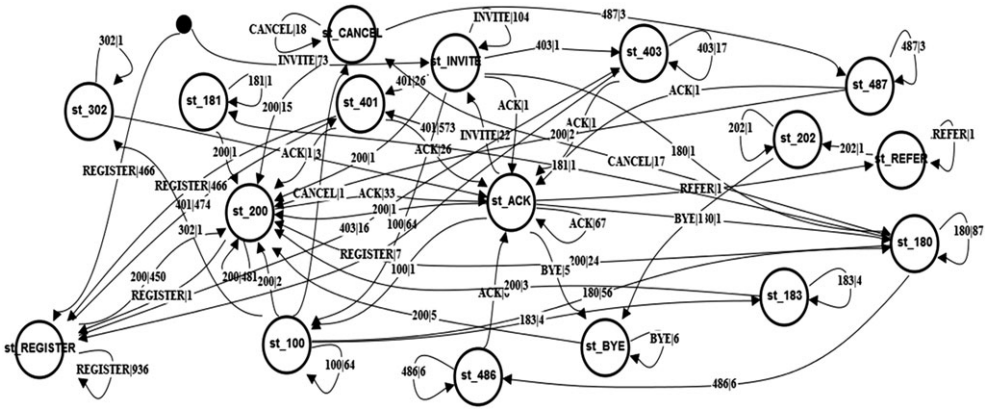

FIGURE 10 Created finite state machine using 5 minutes of a Voice over Internet Protocol provider's calls

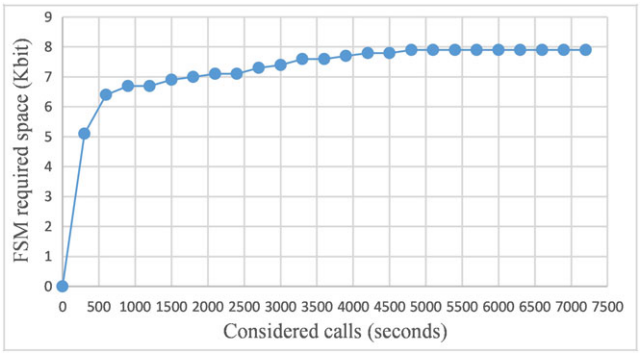

FIGURE 11 Scalability evaluation of the created finite state machine (FSM)

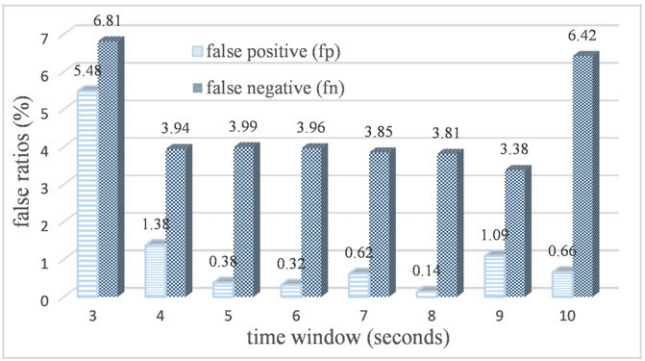

FIGURE 12 Calculated false ratios considering different time windows

used flooding attack from a single source to calculate these ratios. A SIPp attack scenario is used for making calls with the rate of 50 calls per seconds (cps) from one source.

Given that the time window size less than 3 seconds causes an increase in the system resource usage and more than 10 seconds is not acceptable for attack detection, we have considered time interval between 3 to 10 seconds for our experiments. Regarding the obtained results, the best size for the time window of the system is considered equal to 8 seconds.

5.2.1 | Message flow tampering detection

In this case, we considered two kinds of attackers: (1) nonsmart attacker, which initiates calls without considering in progress calls, and (2) smart attacker, which sniffs session parameters of the legitimate users and initiates calls on behalf

of them. In each experiment, the attack traffic was combined with Spirent-generated normal traffic and the number of reached messages to the server was calculated after applying the prevention mechanism. In addition, we compared the results obtained by our system with those of SIPAD method.[8]

To generate nonsmart attacker traffic, we applied a scenario for generating OPTIONS messages with different rates using SIPp tool. To create this scenario, we used nonregistered users. The obtained results are shown in Figure 13. Since SIPAD method uses non-INVITE state transition for non-INVITE messages, it accepts all of the OPTIONS messages regardless of their sessions. Whereas, in standard condition, a user cannot initiate sending OPTIONS messages before registration. Therefore, detecting such attacks is not possible in SIPAD method and, accordingly, almost all OPTIONS messages will reach the server. For generating smart attacker traffic, we used a scenario to make INVITE messages from registered users. As shown in Figure 14, our proposed method can detect more out-of-sequence INVITE messages in comparison with SIPAD method. As previously stated, there is a possibility that a message is observed in several states, which in turn leads to system inability for detection of out-of-sequence messages. As an example, for BYE message, among the 77 observed state transitions in FSM, five cases were due to the BYE message reception. Thus, if an out-of-sequence BYE message is received in one of the 72 remained states, it will correctly be considered as an out-of-sequence one. Hence, an out-of-sequence BYE message can properly be detected by the probability of 93.5%.

The obtained results show that our proposed method is capable of detecting such attacks more accurately in comparison with SIPAD. This difference (shown in Figure 15) is due to the considered states in these two methods. As can be seen, there are only four states in INVITE server state transition model that can receive more than one message, while we considered one state for each message type. The horizontal axis values represent the proposed system state transitions. As a result, the possibility of detecting attack messages will decrease.

5.2.2 | SIP message flooding detection

Single source and distributed flooding attacks

Since the proposed countermeasure method for these two attacks is the same, we investigated the results of the distributed one. To generate attack traffic, we applied a SIPp scenario to make calls for 150 to 180 seconds by defining 150 malicious users along with Spirent-generated normal traffic. In Figure 16, attack traffic with the rate of 60 cps along

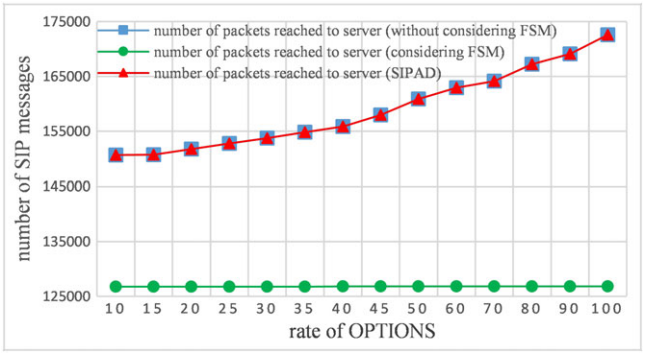

FIGURE 13 Detecting out-of-sequence messages (non-smart attacker)

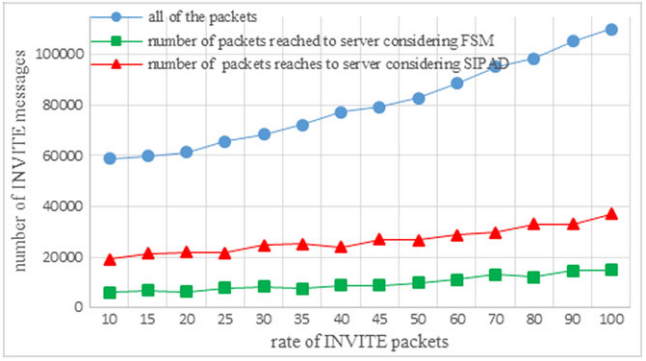

FIGURE 14 Detecting out-of-sequence messages (smart attacker)

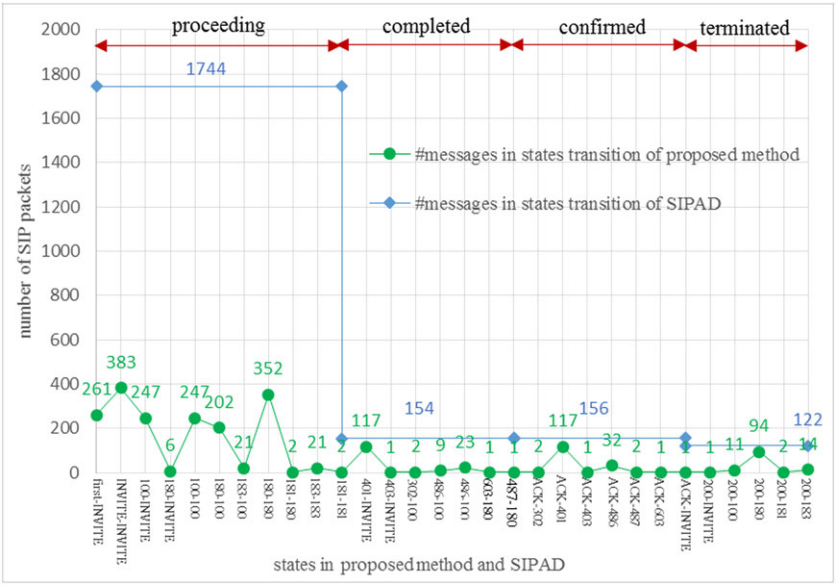

FIGURE 15 The number of observed messages in different states of both methods

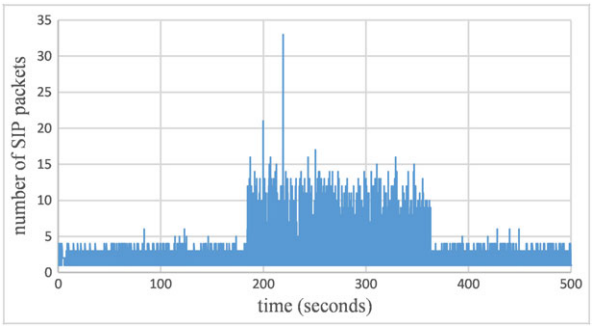

FIGURE 16 Distributed flooding attack with the rate of 60 cps

with normal traffic is presented. Using the same procedure, we generated several attack traffics with different rates. After injecting such traffics to the proposed countermeasure system, we calculated the number of messages reached the server. In Figure 17, the number of SIP messages that reach the server, when there is flooding attack with the rate of 60 cps, is presented.

The obtained results show that almost all attack messages are prevented from reaching the server, but there are still calculable false ratios. To investigate detection accuracy of our proposed method and to compare it with that of SIPAD method, we computed detection probability. This parameter is considered as the percentage of the successfully identified attack samples over the total launched attacks. After generating attack calls with different rates and computing detected attacks at the end of each time window, the detection probability was calculated (Figure 18). Detection probability of our method has a significant increase, especially at lower rate attacks, in comparison with SIPAD method.

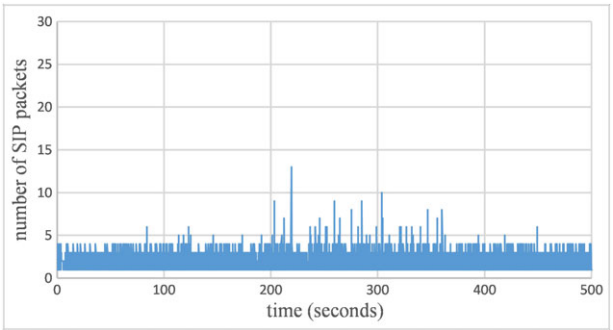

FIGURE 17 Obtained result of detecting and preventing distributed flooding attack (60 cps)

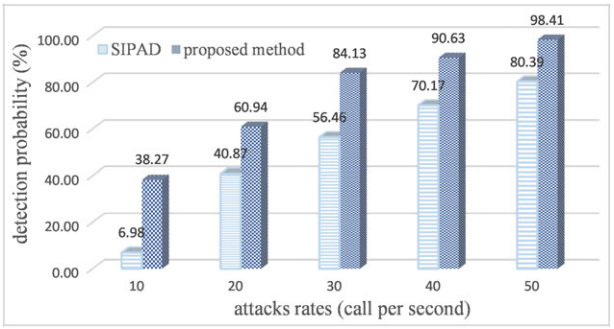

FIGURE 18 Comparison of detection probability in different rates of attack

While the highest and the lowest detection probabilities in our method are equal to 98.41% (at the rate of 50cps) and 38.27% (at the rate of 10cps), respectively; these values are more lower (80.39% and 6.98% at the same rates) in SIPAD method. The reason is that only threshold values are considered for attack detection in SIPAD method, which shows that lower rate attacks cannot be detected accurately.

We calculated false positive and false negative ratios for both methods, considering attacks with different rates. These ratios are shown in Figures 19 and 20, respectively. As can be seen, in comparison with SIPAD method with lower detection probability, our proposed method has better results in almost all attacks with different rates (10cps-60cps). Except at the rate of 10 cps, the lowest rate attack that the gained false positive ratio in our method is greater than SIPAD one (6.31% and 5.61%, respectively), our method has gotten lower results at the other rates (Figure 19). As far as false negative ratios are concerned, both methods have the highest ratios at the rate of 20 cps (Figure 20). Thus, fewer attack messages reached the server, and less normal messages were prevented.

Session flooding attacks

Two different scenarios were considered to launch session flooding attacks: (1) a scenario for generating calls by increasing rate using rate_increase command of SIPp and (2) a scenario for generating sessions without final ACK message

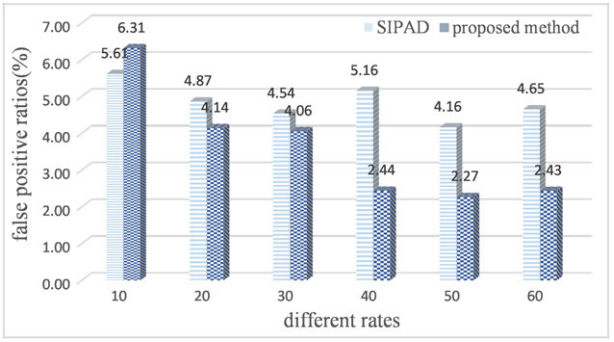

FIGURE 19 False positive ratios for different rates of attack in both methods

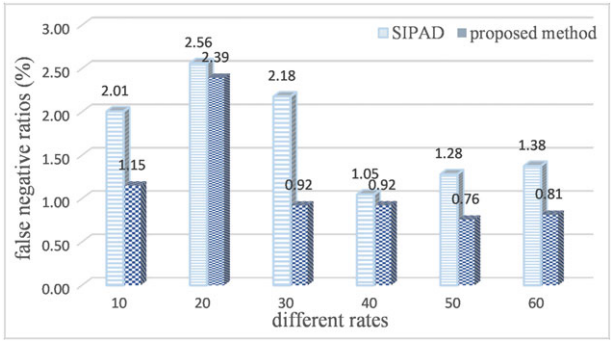

FIGURE 20 False negative ratios for different rates of attack in both methods

(open session). In Figure 21, we launched an increasing rate attack so that in each 10 seconds its rate increases 15 cps and the maximum rate is considered equal to 100 cps. An example of the session without final ACK attack using a scenario with the rate of 30 cps is shown in Figure 22.

To detect these two kinds of flooding attacks, we used Equation (1). By receiving each message and at the end of each time window, we calculated the equation and plotted the obtained results for each second. By increasing the rate of calls, the calculated Session Distance is also increased (due to the number increase of received INVITE requests to the corresponding ACKs). In Figure 23, the calculated Session Distances using both methods are shown. To detect such attacks, we defined a threshold value. Since we expected this proportion be about 1 in normal conditions, therefore, we considered the minimum and maximum threshold values equal to 1 and twice the normal condition value, respectively, to compute a more accurate threshold. Next, by generating different values in this range and investigating results of detection, we picked the best threshold value (1.6). The attack will be detected when the obtained value exceeds the threshold. The plotted results in Figure 23 show that both methods yield approximately the same results, but the SIPAD method may detect such attacks more quickly since it uses 200 OK messages to compute the Session Distance.

In Table 2, detection probability of increasing rates flooding attacks is shown. Both methods can detect such attacks by high probability and accuracy. According to the results, only at the last scenario (rate increase as 15 calls per

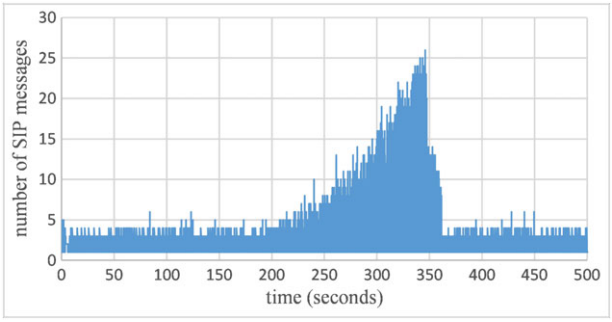

FIGURE 21 Attack traffic with increasing rate (increase 15 calls each 10 seconds)

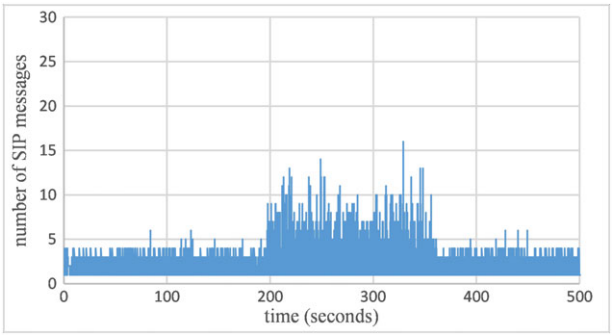

FIGURE 22 Attack traffic using open sessions

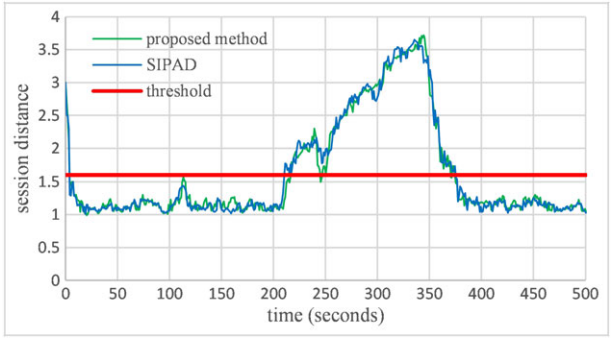

FIGURE 23 Session distance comparison in increasing rate flooding attack

TABLE 2 The detection probability of the increasing rate flooding attack using different rates

	Detection Probability	
Attack Rates	**Proposed Method, %**	**SIPAD, %**
Rate increase as 5 calls per 10 s	87.10	85.48
Rate increase as 10 calls per 10 s	90.32	88.71
Rate increase as 15 calls per 10 s	88.71	91.94

10 seconds), detection probability of the SIPAD (91.94%) is greater than the proposed method (88.71%). Based on the obtained results for Session Distances (Figure 23), these results were predictable.

Moreover, we calculated false ratios after attack detection at the end of each time window (Table 3). Since attack detection possibilities, due to the obtained Session Distances, are close to each other, the possibility of being in Attack state and therefore the resulted false ratios are almost the same.

For creating incomplete sessions, we made traffics with different rates by the help of our modified SIPp scenario. We applied the scenario with the rate of 50 cps and computed Session Distance for both methods the same as before (Figure 24). For calculating an exact number of existing sessions in the server, after the expiration of the prescribed timer, the allocated space is released, and consequently, we deleted the corresponding INVITE requests from the equation. Therefore, by injecting this traffic, the Session Distance increases at first, but a relative drop is inevitable. Like the previous attack, we used threshold value for attack detection. Since the SIPAD method uses 200 OK messages for computing Session Distance, it might not efficiently detect such attacks. As shown in Figure 24, unlike our method, the calculated Session Distance in the SIPAD method is lower than the specified threshold for approximately most of the attack intervals.

The same as previous, we obtained detection probability for different rates (10-50 cps) at the end of each time window. The results (Figure 25) indicate that the possibility of attack detection in the proposed method is significantly higher than that of SIPAD method. While the highest obtained value is 93.55% at the rate of 50 cps in our method, it is equal to 37.1% at the same rate in SIPAD method. However, the lowest value is equal to 80.65% and 16.13% at the rate of 10 cps in the proposed and SIPAD methods, respectively. This is the direct result of the obtained Session Distances for both methods.

After attack detection for each defined attack rate, we calculated false positive and false negative ratios, shown in Figures 26 and 27, respectively. The false positive ratios at all rates (especially at lower rates) in the proposed method

TABLE 3 False ratios in increasing rates flooding attacks

	False Negative		False Positive	
Attack Rates	**Proposed Method, %**	**SIPAD, %**	**Proposed Method, %**	**SIPAD, %**
Rate increase as 5 calls per 10 s	2.43	2.51	5.04	5.10
Rate increase as 10 calls per 10 s	3.38	3.23	4.09	4.14
Rate increase as 15 calls per 10 s	2.97	2.73	3.67	3.97

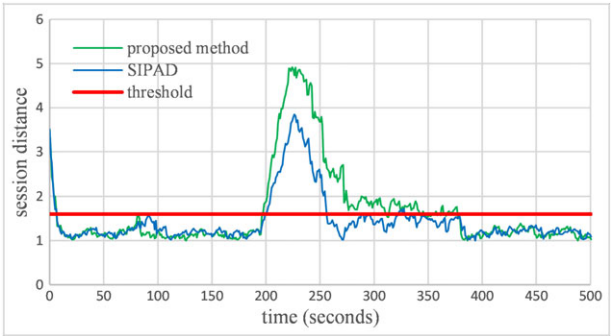

FIGURE 24 Session distance comparison in incomplete session flooding attack (without ACK)

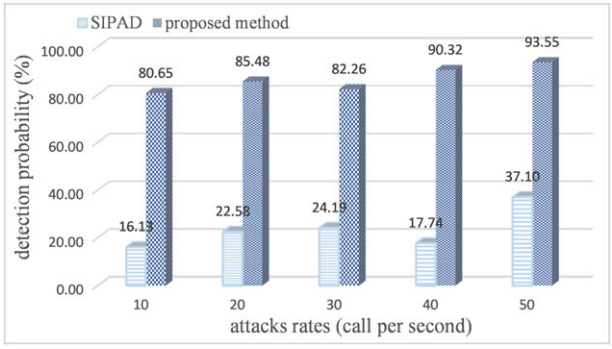

FIGURE 25 Comparison of attack detection probability in attacks with different rates

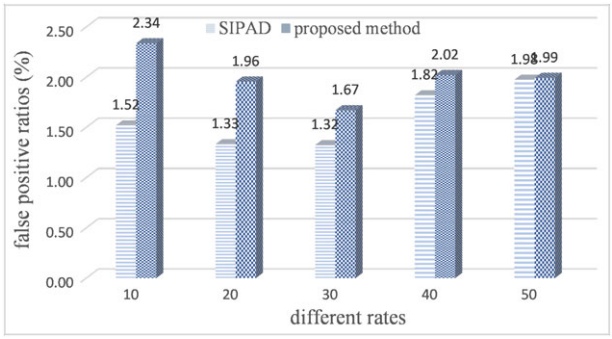

FIGURE 26 False positive ratios for different rates attacks in both methods

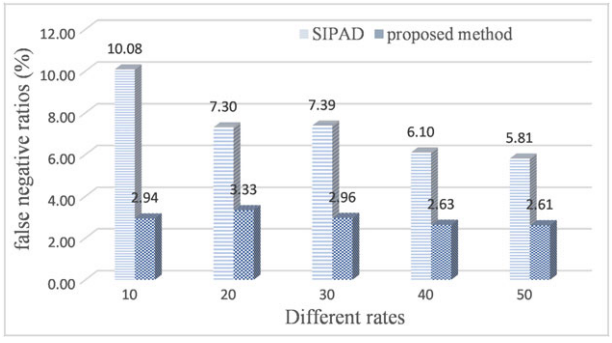

FIGURE 27 False negative ratios for different rates attacks in both methods

are more than SIPAD ones (Figure 26). The reason is that SIPAD is not able to detect such attacks perfectly; therefore, it will place at Attack state less than our method; thus, its false positive ratios are less than those of our method. However, the obtained false negative ratios are completely opposite (Figure 27). The highest ratio in our method is equal to 3.33% (at the rate of 20cps), whereas this ratio is 10.08% in SIPAD method (at the rate of 10 cps).

5.2.3 | Message payload tampering detection

To check the efficiency of the proposed method in detecting malformed attacks, we defined three different scenarios using PROTOS test tool considering: (1) only test cases with an error in the Request-Line field, (2) only test cases with errors in their header fields, and (3) all of the available PROTOS test cases.

The developed structure for detecting malformed messages was compared with SIPAD one. To indicate the ability of the methods to detect incorrect cases and false positive ratios, the results of the experiments are presented in Table 4. The obtained results show that unlike our method, due to the lack of Request-Line checking in SIPAD method,

TABLE 4 Malformed message detection results

	Number of Detected Malformed Messages		False Positive Ratio	
Scenario	**SIPAD, %**	**Proposed Method, %**	**SIPAD, %**	**Proposed Method, %**
Scenario 1	33.13	100	7.14	3.23
Scenario 2	100	100	1.54	1.6
Scenario 3	100	100	4.7	4.03

detecting errors in this part is impossible. Such attacks can only be detected if the malformed field has a similar rule in other header fields, like URI address, and therefore approximately 33.13% of these malformed messages are detectable. In addition, these errors can be detected more quickly in the proposed method because of the fewer rule searches.

5.2.4 | Performance evaluation

In this section, we evaluated the impact of the proposed system on performance of the VoIP system.

Time complexity

For evaluating time complexity of the proposed system, we need to assess time complexity of the different parts of the system. To check the correctness of the message sequences, only some IF-THEN rules were used; therefore, it is required to evaluate time complexity of the fuzzy systems and the tree-based hierarchical structure for detecting flooding attacks and malformed messages, respectively. As regard the performed calculations in Seo et al,[8] the time complexity of detecting malformed messages using the tree-based structure is logarithmic. Thus, we must assess time complexity of the fuzzy systems. As mentioned earlier, there are some fuzzy systems that only some of them are executed according to the kind of received messages at the end of each time window. To evaluate time complexity of each fuzzy system, it is required to evaluate the complexity of the fuzzification, deduction, and defuzzification operations. According to the calculations in Kim et al,[48] each of these operations complexity is constant. Thus, time order of this part depends on the number of received messages at the end of each time window that leads to the FSM state transition. The time order can be considered as $c.\ n$, that c is a constant value equal to the complexity of each fuzzy system and n specifies the number of observed messages, ie, the number of executed fuzzy systems at the end of each time window. Figure 28 illustrates the elapsed time for execution of fuzzy system considering different number of messages at the end of each time window. The obtained results show that time order of these system's execution follows the linear one. It is worth mentioning that the observed messages at the end of each specified time window are normally less than 10 kinds of messages.

Startup and tear-down delays

Since in the proposed method we investigate any kind of SIP messages to detect attacks, some unwanted delays is imposed on the VoIP system, whether during a new call startup or when a call tears down. To examine how much our proposed mechanism imposes delay to the system, we run flooding attack traffic. The amount of delay imposed on the system during call startup and call tear-down is shown in Figures 29 and 30, respectively. The obtained results

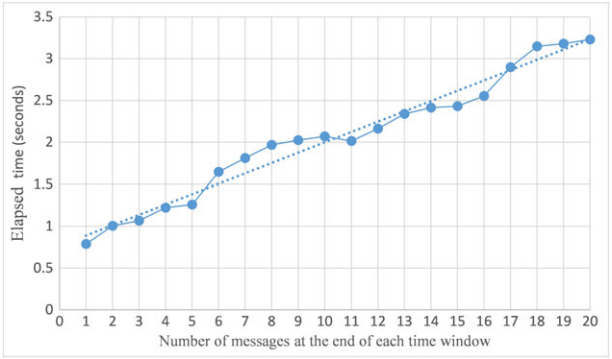

FIGURE 28 The time complexity of the fuzzy system's execution

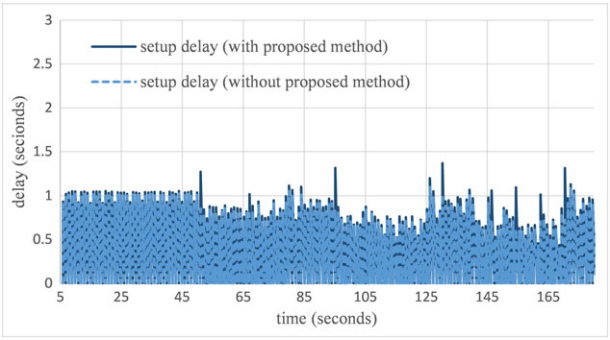

FIGURE 29 The calculated startup delay

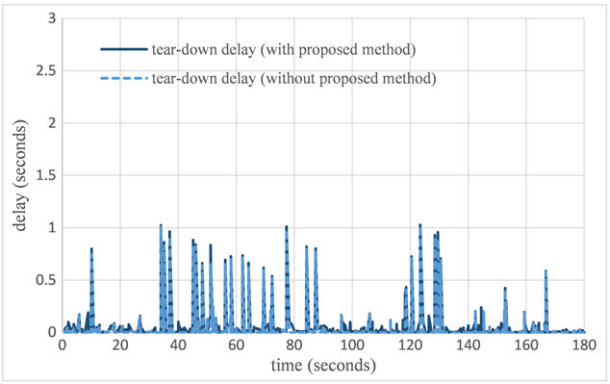

FIGURE 30 The calculated tear-down delay

show that on average, our method imposes between 3% and 7% of the delay during call startup. However, the delay imposed during call tear-down is about 4%.

We also compared our proposed method with the method presented in Roh et al[32] as a function of throughput. The authors of that paper defined the throughput as the ratio between the number of INVITEs sent from normal users and the number of INVITEs received by the server. In that study, the authors used a whitelist method simulated by Bloom filter and a nonmembership ratio to detect attack situations. We combined the normal and attack traffic generated by Spirent device and SIPp tool, respectively, to compute throughput for both methods. In our attack scenario, we defined 100 new registered users, which made calls at a low rate (15 cps). The throughput of our system in comparison with the presented method in Roh et al[32] is illustrated in Figure 31. As can be seen, the throughput of our proposed method is much better than Roh et al method and most of the time is equal to 1. The main reason for this difference is that calls generated from new registered users, leads to an increase in nonmembership ratio and in turn to become greater than threshold value. Therefore, the attack symptom is detected and INVITE messages that are not in the whitelist are prevented from reaching the server. However, in our method, only in some time windows, attack symptoms are wrongly detected and INVITE messages which are not in the whitelist are prevented.

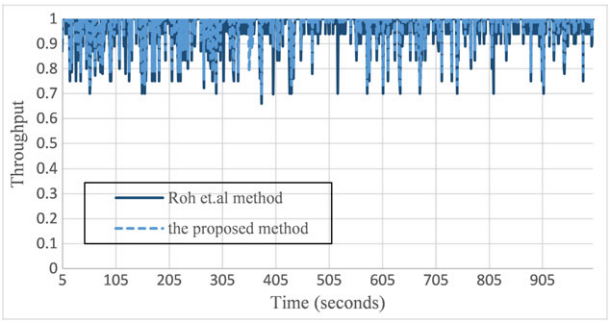

FIGURE 31 Throughput comparison

6 | CONCLUSION

SIP-based multimedia systems such as VoIP services are becoming so popular for communication purposes. However, these services are exposed to some kinds of attacks such as DoS attacks due to SIP flow tampering, SIP payload tampering, and flooding different kinds of SIP messages. In this paper, a new method is presented for detecting and preventing such attacks. To reach this goal, a FSM is used for modeling SIP normal traffic of a VoIP provider. Using extracted parameters in this phase, the correct sequence of SIP messages can be investigated. Fuzzy systems are applied in order to detect flooding attack due to numerous SIP messages. Furthermore, a new equation is proposed to detect open sessions during the test period. A tree-based hierarchical structure is presented to detect malformed messages by comparing with the true syntax of the messages. At this point and after detecting the attack, the system will be placed at a predefined state: ie, Normal, Alarm, and Attack states. To prevent attacker messages reaching the server in Attack state, a prevention mechanism using a whitelist-based method is proposed. For considering space efficiency, the whitelist is made by the help of a Bloom filter mechanism. This method is completely implemented, and the obtained results show its higher accuracy and less false positives and false negative alarms while imposing less delay in comparison with a similar method.

ORCID

Mohammad Hossein Yaghmaee http://orcid.org/0000-0002-0111-1357

REFERENCES

1. Geneiatakis D, Dagiuklas T, Kambourakis G, et al. Survey of security vulnerabilities in Session Initiation Protocol. *IEEE Commun Surv Tutorials*. 2006;8(1-4):68-81.
2. Ehlert S, Geneiatakis D, Magedanz T. Survey of network security systems to counter SIP-based denial-of-service attacks. *Comput Secur*. 2010;29(2):225-243.
3. Rosenberg J, Schulzrinne H, Camarillo G, et al. SIP: Session Initiation Protocol 2002.2070-1721.
4. Keromytis AD. A comprehensive survey of voice over IP security research. *IEEE Commun Surv Tutorials*. 2012;14(2):514-537.
5. Chen EY, Itoh M. A whitelist approach to protect SIP servers from flooding attacks. Paper Presented at: 2010 IEEE International Workshop Technical Committee on Communications Quality and Reliability (CQR 2010)2010.
6. Butcher D, Li X, Guo J. Security challenge and defense in VoIP infrastructures. *IEEE Trans Syst Man Cybern Part C Appl Rev*. 2007;37(6):1152-1162.
7. Geneiatakis D, Vrakas N, Lambrinoudakis C. Utilizing bloom filters for detecting flooding attacks against SIP based services. *Comput Secur*. 2009;28(7):578-591.
8. Seo D, Lee H, Nuwere E. SIPAD: SIP–VoIP anomaly detection using a stateful rule tree. *Comput Commun*. 2013;36(5):562-574.
9. Salsano S, Veltri L, Papalilo D. SIP security issues: the SIP authentication procedure and its processing load. *IEEE Netw*. 2002;16(6):38-44.
10. Voznak M, Safarik J. DoS attacks targeting SIP server and improvements of robustness. *Int J Math Comput Simul*. 2012;6(1):177-184.
11. Ehlert S, Zhang G, Geneiatakis D, et al. Two layer Denial of Service prevention on SIP VoIP infrastructures. *Comput Commun*. 2008;31(10):2443-2456.
12. Vrakas N, Geneiatakis D, Lambrinoudakis C. Evaluating the security and privacy protection level of IP multimedia subsystem environments. *IEEE Commun Surv Tutorials*. 2013;15(2):803-819.
13. Armoogum S, Mohamudally N. Survey of practical security frameworks for defending SIP based VoIP systems against DoS/DDoS attacks. Paper Presented at: IST-Africa Conference Proceedings, 20142014.
14. Kumar A, Tilagam S. A novel approach for evaluating and detecting low rate SIP flooding attack. *Int J Comput Appl*. 2011;26(1):31-36.
15. Wu Y-S, Bagchi S, Garg S, Singh N. Scidive: a stateful and cross protocol intrusion detection architecture for voice-over-ip environments. Paper Presented at: Dependable Systems and Networks, 2004 International Conference on2004.
16. Rezac F, Voznak M, Tomala K, Rozhon J, Vychodil J. Security analysis system to detect threats on a SIP VoIP infrasctructure elements. *Adv Electr Electron Eng Ser*. 2011;9(5):225.
17. Hussain I, Nait-Abdesselam F. Strategy based proxy to secure user agent from flooding attack in SIP. Paper Presented at: Wireless Communications and Mobile Computing Conference (IWCMC), 2011 7th International2011.
18. Lahmadi A, Festor O. A framework for automated exploit prevention from known vulnerabilities in voice over IP services. *IEEE Trans Netw Serv Manag*. 2012;9(2):114-127.

19. Ormazabal G, Nagpal S, Yardeni E, Schulzrinne H. Secure sip: a scalable prevention mechanism for dos attacks on sip based voip systems. In: *Principles, systems and applications of IP telecommunications Services and security for next generation networks*. Springer; 2008:107-132.
20. Li H, Lin H, Yang X, Liu F. A rules-based intrusion detection and prevention framework against SIP malformed messages attacks. Paper presented at: Broadband Network and Multimedia Technology (IC-BNMT), 2010 3rd IEEE International Conference on2010.
21. Lu J, Dousson C, Krief F. Distributed self-diagnosis algorithm for VoIP service over IMS network. Paper Presented at: Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2015 International Conference on2015.
22. De Rango F, Fazio P, Scarcello F, Conte F. A new distributed application and network layer protocol for voip in mobile ad hoc networks. *IEEE Trans Mob Comput*. 2014;13(10):2185-2198.
23. Reynolds B, Ghosal D. Secure IP Telephony using Multi-layered Protection. Paper presented at: NDSS2003.
24. Sengar H, Wijesekera D, Wang H, Jajodia S. VoIP intrusion detection through interacting protocol state machines. Paper presented at: International Conference on Dependable Systems and Networks (DSN'06)2006.
25. Rebahi Y, Sher M, Magedanz T. Detecting flooding attacks against IP multimedia subsystem (IMS) networks. Paper Presented at: 2008 IEEE/ACS International Conference on Computer Systems and Applications2008.
26. Li H, Lin H, Hou H, Yang X. An efficient intrusion detection and prevention system against sip malformed messages attacks. Paper Presented at: Computational Aspects of Social Networks (CASoN), 2010 International Conference on2010.
27. Sengar H, Wang H, Wijesekera D, Jajodia S. Detecting VoIP floods using the Hellinger distance. *IEEE Trans Parallel Distrib Syst*. 2008;19(6):794-805.
28. Tang J, Cheng Y, Hao Y. Detection and prevention of SIP flooding attacks in voice over IP networks. Paper Presented at: INFOCOM, 2012 Proceedings IEEE2012.
29. Zhang J, Qin Z, Ou L, Jiang P, Liu J, Liu AX. An advanced entropy-based DDOS detection scheme. Paper presented at: 2010 International Conference on Information, Networking and Automation (ICINA)2010.
30. Tang J, Cheng Y. Quick detection of stealthy sip flooding attacks in voip networks. Paper Presented at: 2011 IEEE International Conference on Communications (ICC)2011.
31. Asgharian Z, Asgharian H, Akbari A, Raahemi B. A framework for SIP intrusion detection and response systems. Paper Presented at: Computer Networks and Distributed Systems (CNDS), 2011 International Symposium on2011.
32. Roh B-h, Kim JW, Ryu K-Y, Ryu J-T. A whitelist-based countermeasure scheme using a Bloom filter against SIP flooding attacks. *Comput Secur*. 2013;37:46-61.
33. Tsiatsikas Z, Geneiatakis D, Kambourakis G, Keromytis AD. An efficient and easily deployable method for dealing with DoS in SIP services. *Comput Commun*. 2015;57:50-63.
34. Tsiatsikas Z, Geneiatakis D, Kambourakis G, Keromytis AD. A privacy-preserving entropy-driven framework for tracing DoS attacks in VoIP. Paper Presented at: Availability, Reliability and Security (ARES), 2013 Eighth International Conference on2013.
35. Marchal S, Mehta A, Gurbani VK, State R, Ho TK, Sancier-Barbosa F. Mitigating mimicry attacks against the session initiation protocol. *IEEE Trans Netw Serv Manag*. 2015;12(3):467-482.
36. Safarik J, Partila P, Rezac F, Macura L, Voznak M. Automatic classification of attacks on IP telephony. *Adv Electr Electron Eng Ser*. 2013;11(6):481.
37. Golait D, Hubballi N. VoIPFD: voice over IP flooding detection. Paper Presented at: 2016 Twenty Second National Conference on Communication (NCC); 4-6 March 2016, 2016.
38. Shah B, Dave K. SIP based intrusion detection system for VoIP based applications. Paper presented at: Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies2016.
39. Cadet F, Fokum DT. Coping with denial-of-service attacks on the IP telephony system. Paper Presented at: SoutheastCon 20162016.
40. Tas IM, Ugurdogan B, Baktir S. Novel Session Initiation Protocol -based distributed denial-of-service attacks and effective defense strategies. *Comput Secur*. 2016;63:29-44.
41. Raza MA, Raza M. A restrictive model (RM) for detection and prevention of INVITE flooding attack. Paper presented at: Computer, Control & Communication (IC4), 2013 3rd International Conference on2013.
42. Ghafarian A, Hosseini Seno SA, Dehghani M. An empirical study of security of VoIP system. Paper Presented at: 2016 SAI Computing Conference (SAI); 13-15 July 2016, 2016.
43. Yu K, Li FY, Wu X, Di J. The heterogeneity of inter-domain internet application flows: entropic analysis and flow graph modelling. *Trans Emerg Telecommun Technol*. 2015;26(5):760-771.
44. Antunes J, Neves N, Verissimo P. Reverse engineering of protocols from network traces. Paper Presented at: 2011 18th Working Conference on Reverse Engineering2011.
45. Abaev P, Razumchik R, Uglov I. Statistical analysis and modeling of SIP traffic for parameter estimation of server hysteretic overload control. *Res J Telecommun Inf Technol*. 2013;4:22-31.

46. Chiappetta S, Mazzariello C, Presta R, Romano SP. An anomaly-based approach to the analysis of the social behavior of VoIP users. *Comput Netw*. 2013;57(6):1545-1559.

47. Fan L, Cao P, Almeida J, Broder AZ. Summary cache: a scalable wide-area web cache sharing protocol. *IEEE/ACM Trans Networking*. 2000;8(3):281-293.

48. Kim YH, Ahn SC, Kwon WH. Computational complexity of general fuzzy logic control and its simplification for a loop controller. *Fuzzy Set Syst*. 2000;111(2):215-224.

How to cite this article: Hosseinpour M, Yaghmaee MH, Hosseini Seno SA, Khosravi Roshkhari H, Asadi M. Anomaly-based DoS detection and prevention in SIP networks by modeling SIP normal traffic. *Int J Commun Syst*. 2018;e3825. https://doi.org/10.1002/dac.3825