available at www.sciencedirect.com

journal homepage: www.elsevier.com/locate/cose

Computers & Security

Utilizing bloom filters for detecting flooding attacks against SIP based services

Dimitris Geneiatakis[*], *Nikos Vrakas*, *Costas Lambrinoudakis*

Laboratory of Information and Communication Systems Security, Department of Information and Communication Systems Engineering, University of the Aegean, Karlovassi, GR-83200 Samos, Greece

ARTICLE INFO

Article history:
Received 12 August 2007
Received in revised form
26 April 2009
Accepted 27 April 2009

Keywords:
Session Initiation Protocol (SIP)
Voice over IP (VoIP)
Flooding attacks
Denial of Service
Bloom filter
Security

ABSTRACT

Any application or service utilizing the Internet is exposed to both general Internet attacks and other specific ones. Most of the times the latter are exploiting a vulnerability or misconfiguration in the provided service and/or in the utilized protocol itself. Consequently, the employment of critical services, like Voice over IP (VoIP) services, over the Internet is vulnerable to such attacks and, on top of that, they offer a field for new attacks or variations of existing ones. Among the various threats–attacks that a service provider should consider are the flooding attacks, at the signaling level, which are very similar to those against TCP servers but have emerged at the application level of the Internet architecture. This paper examines flooding attacks against VoIP architectures that employ the Session Initiation Protocol (SIP) as their signaling protocol. The focus is on the design and implementation of the appropriate detection method. Specifically, a bloom filter based monitor is presented and a new metric, named *session distance*, is introduced in order to provide an effective protection scheme against flooding attacks. The proposed scheme is evaluated through experimental test bed architecture under different scenarios. The results of the evaluation demonstrate that the required time to detect such an attack is negligible and also that the number of false alarms is close to zero.

1. Introduction

The World Wide Web (WWW) has been designed to provide a mean for sharing information among specific communities, utilizing a ‘‘unified’’ open network as the Internet. At the early stages of Internet deployment no security considerations were taken into account. This fact has been exploited in several ways by many malicious users who have caused numerous incidents, one of them being Denial of Service (DoS) (i.e. they have stopped illegally the communication among the communicating parties). By the term of DoS is identified any attempt by malicious users trying to constitute the provided service unavailable to legitimate users. The existence of such a problem was firstly pinpointed in Gligor (1984), in which the malicious users focus on the vulnerabilities of the operating system. Generally DoS attacks could have two major forms. In the first one, the malicious user crafts very carefully a packet trying to exploit vulnerabilities in the implemented software (service or a protocol). Among the most known attacks of this category are the buffer overflow attacks while an incident of such an attack is the Ping of Death. In the second form, the malicious user is trying to overwhelm system's resources of the provided service-like memory, CPU or bandwidth, by creating numerous of useless well-formed requests. This type of attack is well known as flooding attack. The most known flooding attack against web servers is reported in Gibson

[*] *Corresponding author*. Tel.: +30 22730 82247; fax: +30 22730 82009.
E-mail addresses: dgen@aegean.gr (D. Geneiatakis), nvrak@aegean.gr (N. Vrakas), clam@aegean.gr (C. Lambrinoudakis).

0167-4048/$ – see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cose.2009.04.007

(2002). A detailed analysis of DoS attacks in Internet services can be found in Carl et al. (2006), Peng et al. (2007) and Mirkovic et al. (2004).

Consequently, any application and/or service utilizing the Internet's open architecture is constituted susceptible to similar attacks. It is therefore reasonable for every critical real-time application to treat architectures like the Internet as hostile environments. This is, or at least should be, the case for Voice over IP (VoIP) services offered over the Internet. Furthermore, various researchers have already identified security vulnerabilities in those systems (Endler et al., 2005; Sisalem et al., 2005; Geneiatakis et al., 2006; Ming et al., 2008). On the contrary Public Switch Telephone Network (PSTN) security flaws are considered minimal, due to its closed network architecture. Consequently, in PSTN it is considered very difficult to launch any kind of attack provided that the attacker has no physical access to the network. This is not the case for VoIP based Internet services, since a vulnerability in the service/protocol can be exploited by utilizing various VoIP security tools like SIP swiss army knife (sipsak), PROTOS testing suite (Wieser et al., 2003) or by developing specific tools as demonstrated in Endler et al. (2005), Sisalem et al. (2005) and Geneiatakis et al. (2006), without requiring physical access to the medium. Additionally, a side effect of the interconnection between VoIP and PSTN is that PSTN infrastructures become vulnerable to VoIP attacks. Consider, for instance, that a VoIP service is used as an intermediate for launching a flooding attack against the PSTN infrastructure. Such a scenario was not feasible a few years ago.

It is therefore clear that the protection of VoIP systems against flooding attacks is crucial not only to ensure that the trust and security levels offered are similar to those offered by PSTN architectures, but also to protect the PSTN system itself against flooding attacks. In the work presented here we examine the case of flooding attacks against VoIP systems that employ the Session Initiation Protocol (SIP) (Rosenberg et al., 2002) for call management. For protecting SIP proxies' servers against flooding attacks, we propose a monitor system that is based on bloom filters combined with a new metric, named *session distance*. Our main focus is on SIP as it seems to overwhelm the other signaling protocols (H.323, MGCP) and it has been adopted by various standardization (e.g. 3GPP, IETF) organizations as the protocol to establish multimedia sessions at both wireline and wireless world in the Next Generation Networks (NGN) era.

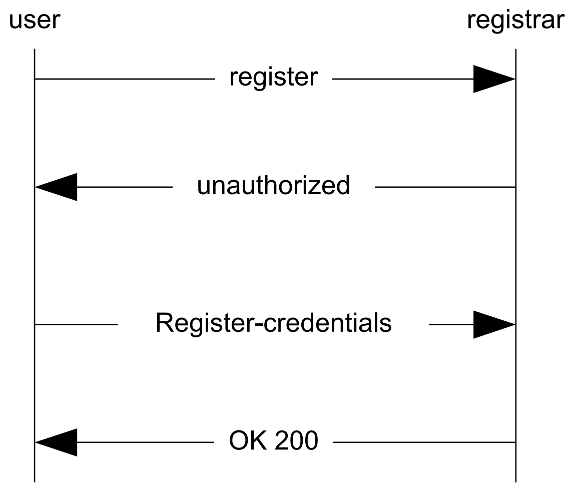
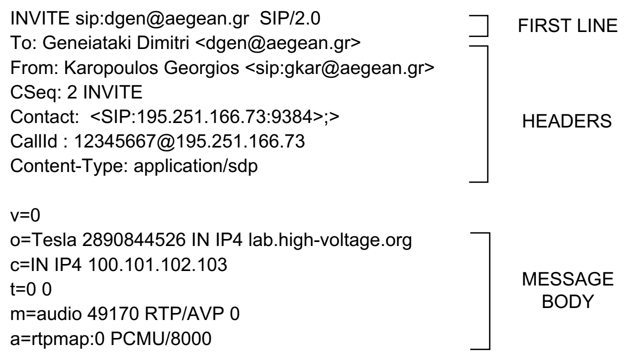


Fig. 2 – Registration procedure.

The rest of the paper is structured as follows. Section 2 provides background information on the SIP protocol. Section 3 presents various types of flooding attacks against SIP based VoIP systems while Section 4 demonstrates their consequences. Section 5 presents and evaluates a novel detection method for flooding attacks, which is based on a *bloom filter*. Finally Section 6 acquaints with related work and Section 7 concludes the paper giving some pointers for future work.

2. The Session Initiation Protocol

Session Initiation Protocol (SIP) is an application-layer signaling protocol for creating, modifying, and terminating multimedia sessions among one or more participants (Rosenberg et al., 2002). The general structure of the SIP protocol is inherited by Hyper Text Transfer Protocol (HTTP) (Fielding et al., 1999) and thus SIP messages are text based similar to the HTTP ones. Specifically, a SIP message can be either a request or the corresponding response, depending on the first line of the message, followed by the appropriate headers required to describe the request or the response, and the message body. The message body is optional and its existence depends on the request. Fig. 1 illustrates an example



```
INVITE sip:dgen@aegean.gr  SIP/2.0
To: Geneiataki Dimitri <dgen@aegean.gr>
From: Karopoulos Georgios <sip:gkar@aegean.gr>
CSeq: 2 INVITE
Contact:  <SIP:195.251.166.73:9384>;>
CallId : 12345667@195.251.166.73
Content-Type: application/sdp

v=0
o=Tesla 2890844526 IN IP4 lab.high-voltage.org
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

FIRST LINE
HEADERS
MESSAGE BODY

Fig. 1 – A typical SIP-INVITE request.

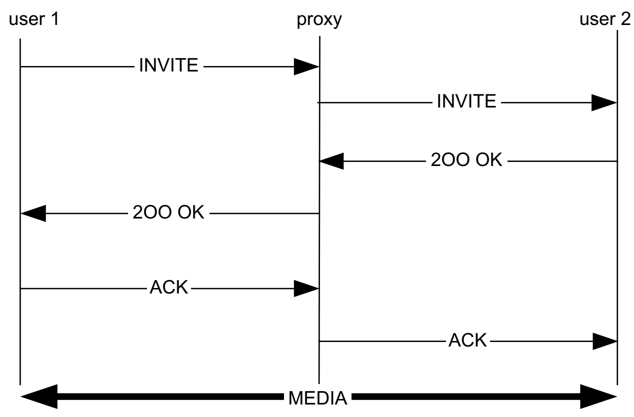


Fig. 3 – SIP multimedia connection establishment.

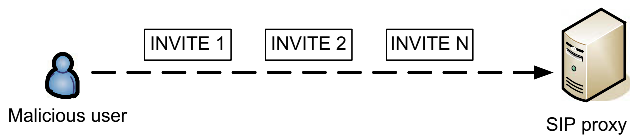


Fig. 4 – An example of single-source flooding against proxy.

of a typical INVITE request. The INVITE request is utilized in order to call one or more participants to a particular session.

It should be noted that the differentiation of a SIP response from the corresponding SIP request is located at the first line, which is replaced with the response code followed by the appropriate description. For instance the response code of an accepted invitation is "200", while the description is "OK". As a result the first line of the response message corresponding to the request shown in Fig. 1 will read "*200 OK SIP*".

Normally a legal user before using a SIP service must employ the registration service, provided by the Registrar server, in order to sign in by generating and sending a REGISTER message that includes her credentials. The SIP registration procedure is illustrated in Fig. 2. It should be stressed that the authentication mechanism relies on HTTP digest (Franks et al., 1999).

Provided that the registration phase has been successfully completed, the user can utilize a service, for example the call service, by contacting the proxy server that is responsible to "find" the requested resources/services.

Let's consider the INVITE request that is employed in order to call other users. As already mentioned, the specific request is directed to the proxy server that is responsible for the administration of the INVITE messages and then for forwarding the generated requests to the appropriate user or resource. The procedure followed when a user 1 (caller) calls a user 2 (callee) is presented in Fig. 3. The caller generates and sends the INVITE request to the appropriate proxy server. Assuming that the callee is available and accepts the call, he generates the "*200 OK*" response message.

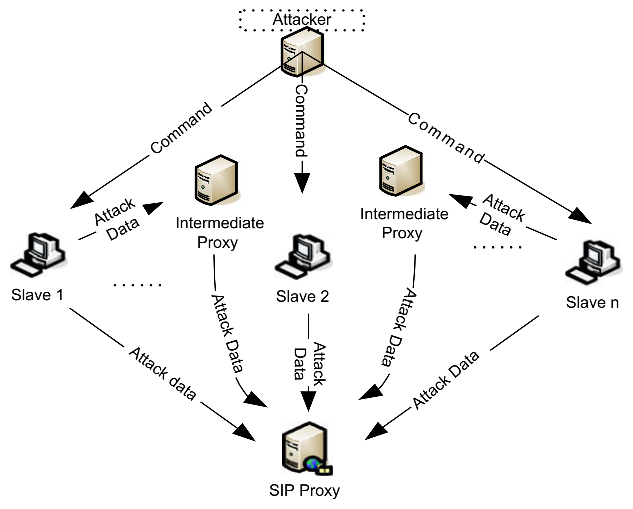


Fig. 5 – Distributed INVITE flooding in SIP.

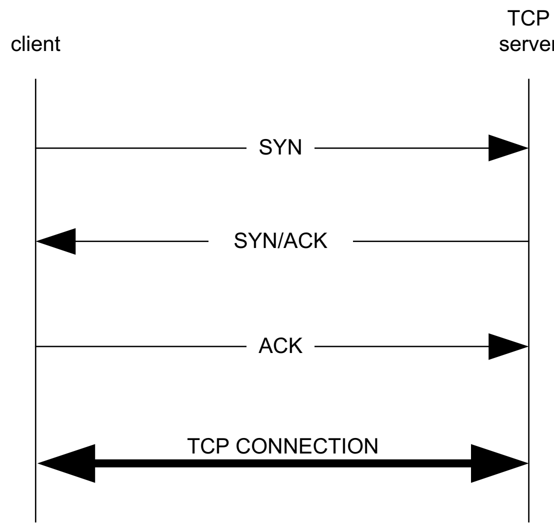


Fig. 6 – A typical 3-way TCP handshake.

The proxy server can operate either in a stateless or in a stateful mode. When acting stateless the proxy simply forwards the incoming request (in our case the INVITE). This means that it simply needs to maintain a copy of the received message until the message has been sent out to the requested destination. On the contrary when the proxy is acting stateful, in addition to the copy of the received request the proxy also needs to maintain the state of the transaction. In other words a stateful proxy needs to keep a copy of the received request as well as a copy of the forwarded request and the corresponding memory for identifying the transaction state. The proxy should maintain the transaction state until a final response is received or a time-out is triggered. Therefore a stateful proxy may maintain the transaction from a few seconds to some minutes, depending on the proxy implementation and configuration. For more details about the SIP protocol the reader could see Rosenberg et al. (2002).

3. Flooding attacks in SIP systems

The main goal of any flooding attack is the consumption of system resources, like bandwidth, memory or/and CPU time,

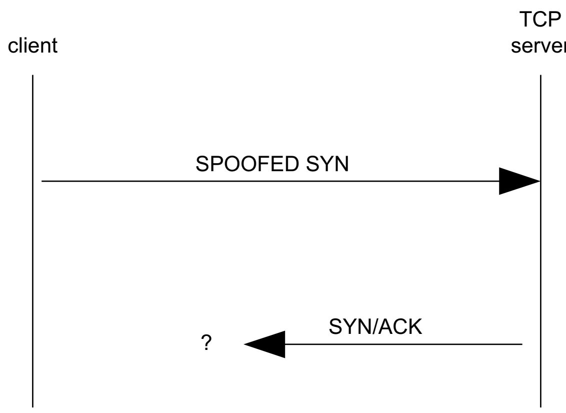


Fig. 7 – TCP-SYN attack.

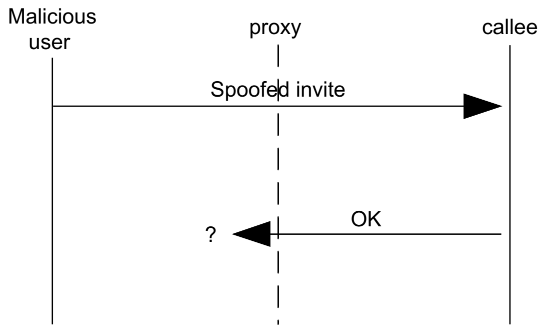


Fig. 8 – SIP-INVITE SYN syndrome attack.

in order to make the service unavailable. SIP based VoIP services are susceptible to the following flooding attacks:

- *Registrar server flooding*: In this case the malicious user tries to cause a DoS to the Registrar server by generating and sending numerous REGISTER messages, forcing the corresponding server to execute expensive cryptographic operations.
- *Proxy server flooding*: In this case the malicious user generates several INVITE messages. A proxy server that operates in a stateful mode is more vulnerable to such flooding attacks due to the fact that it should also administer the state of the transaction of any new request that could last up to several minutes. Thus a stateful proxy consumes more memory and CPU time in order to maintain transactions and new requests, suffering a more rapid resource consumption rate as compared to a stateless one. We have distinguished the INVITE flooding attacks in the following subcategories (more details are presented in Section 3.1):
  - *INVITE single or multiple source attack*
  - *INVITE-SYN syndrome attack*
  - *INVITE-reflection syndrome attack*

These attacks can be launched either by internal or external users, authenticated or not. In cases where an attack is launched by a non-authenticated user, innocent proxies are exploited and used as intermediates in order to cause a DoS to some other SIP proxy. Within the scope of this paper we focus on flooding attacks against proxy servers and not against the Registrar server that is responsible for user authentication.

It should be stated that flooding attacks in SIP are more attractive to malicious users due to the fact that SIP's text based form offers more opportunities to construct SIP spoofed messages. Consequently in all types of a flooding attack, the malicious user generates a spoofed INVITE message by using a string random generator to create the requested resource and the corresponding headers similar to those depicted in Fig. 1. Depending on the target (proxy or end-user) the spoofed message is created accordingly.

3.1. *Flooding attacks against proxy servers*

As already pointed out, the management of incomplete INVITE transactions constitutes the corresponding proxies vulnerable to flooding attacks, as the proxy has to administer the state of the transaction until the release of the allocated resources, something that could take several minutes (see Section 2). The following subsections present in detail the various ways that an INVITE request could be employed by a malicious user in order to degrade SIP based VoIP system resources and finally cause a DoS.

3.1.1. *INVITE single–multiple source flooding attacks*

Consider a malicious user who creates numerous bogus SIP-INVITE requests and forwards them to a proxy in order to

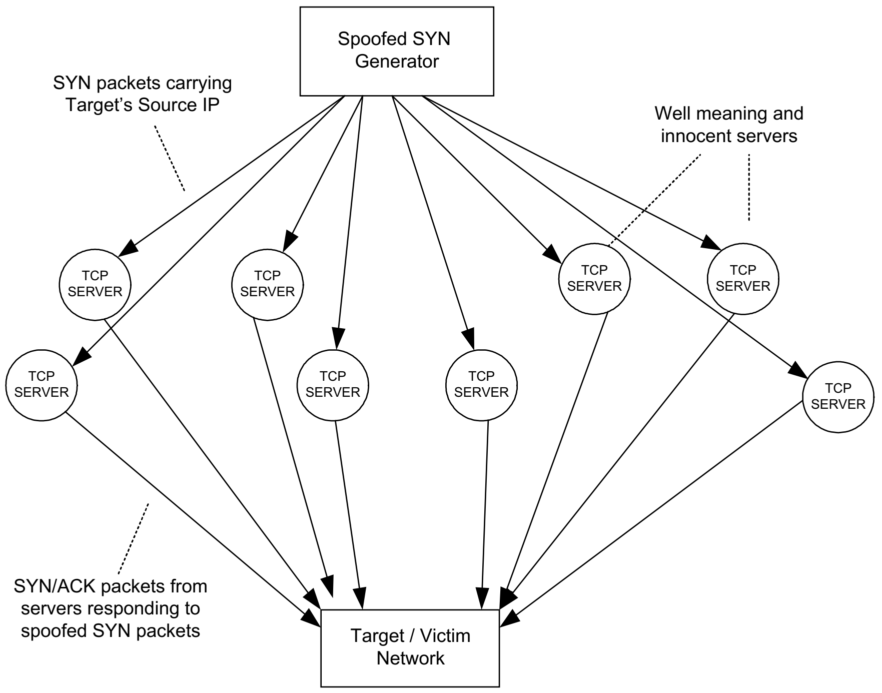

Spoofed SYN Generator
SYN packets carrying Target's Source IP
Well meaning and innocent servers
TCP SERVER
SYN/ACK packets from servers responding to spoofed SYN packets
Target / Victim Network

Fig. 9 – The well known distributed reflection DoS attack.

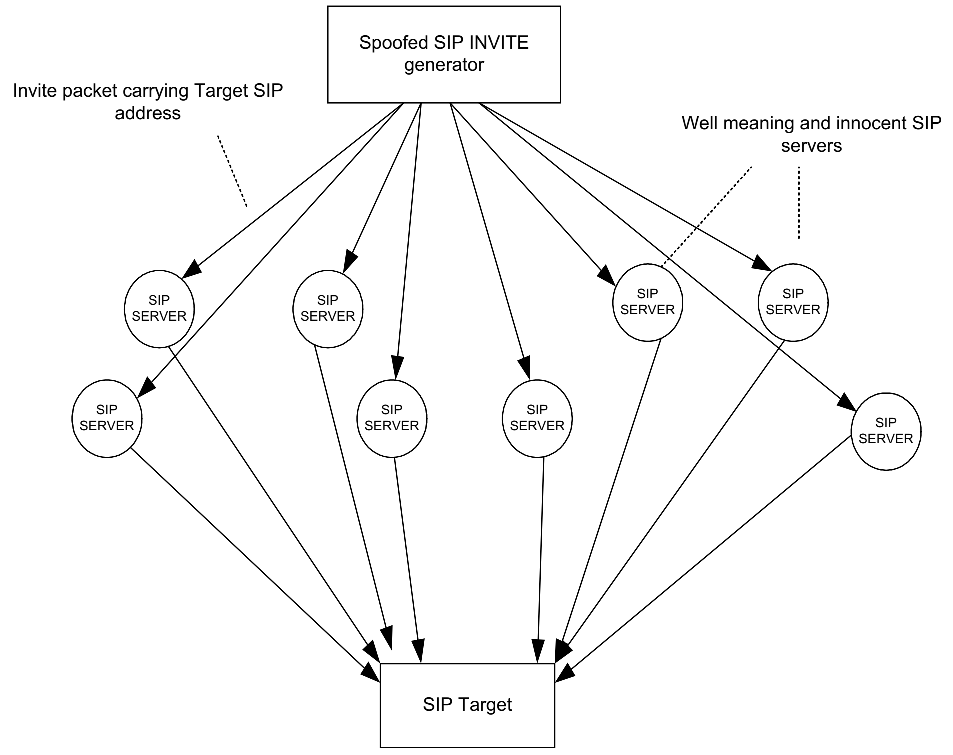

Spoofed SIP INVITE generator
Invite packet carrying Target SIP address
Well meaning and innocent SIP servers
SIP SERVER
SIP Target

Fig. 10 – The INVITE-reflection syndrome.

overwhelm its resources or its inability to handle such a big number of requests, causing a DoS. In such cases the generated INVITE requests must correspond to different sessions. This means that the malicious user crafts messages similar to those presented in Fig. 1, but with a different request *uri*- or *call-id* header for each new INVITE message. In this way the attacker tries to consume memory and CPU resources, as the proxy would allocate memory for each new incoming request that will be only released when a final response is received or a time-out is triggered. Consequently, if the number of requests is big the server memory will be exhausted causing a DoS or its performance will be degraded. An example of such an attack is illustrated in Fig. 4, in which the malicious user sends bogus requests in random intervals.

In order to launch a more powerful attack (causing a more rapid consumption of SIP server's resources) the attacker may compromise various innocent SIP servers giving a distributed nature to the attack similar to the master-slave communication architecture employed in flooding attacks against Internet services. More specifically the attacker sends to the "slaves" (innocent proxies) a specific command, e.g. an INVITE flood, and the slaves start to send INVITE requests (attack-data) against the target-proxy. An example of such an attack in SIP is illustrated in Fig. 5. This type of attack is exactly the same with the single flooding attack with the only difference that the attackers are geographically distributed.

3.1.2. *INVITE-SYN syndrome attacks*

The SIP similarities with Transport Control Protocol (TCP) (Postel, 1981) constitute it vulnerable to similar attacks. Specifically, it is well known that the TCP is one of the main transport protocols employed in Internet to connect geographically distribute remote machines. The establishment of a TCP connection requires a three-way handshake procedure illustrated in Fig. 6.

Even though TCP is utilized for providing communication among distributed machines, malicious users have exploited a vulnerability in the implementation of TCP/IP stack, causing DoS by consuming the corresponding TCP server's resources. A well known attack of this type is the TCP/SYN flooding attack (Center, 1996) that consumes server's memory. Specifically, the attacker sends numerous TCP-SYN requests to the corresponding TCP server. For any new TCP request the server allocates the appropriate memory for handling the connection, and then generates the corresponding TCP-SYN/ACK message. Due to the fact that the TCP/SYN message includes a spoofed IP address the generated SYN/ACK responses do not reference a real system, causing the server to

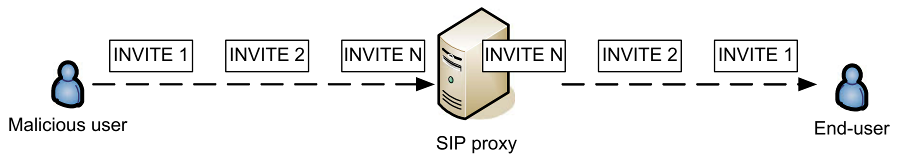


Fig. 11 – The single case of SIP-INVITE flood against an end-user.

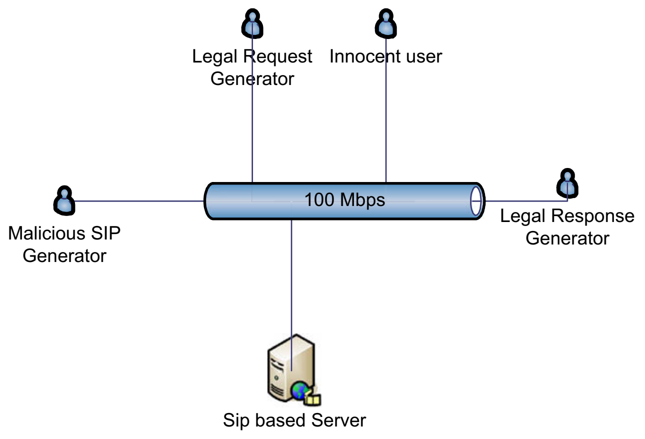


Fig. 12 – Attack architecture for SIP based architecture.

reserve resources until the appropriate TCP timer expires. The TCP-SYN attack procedure is illustrated in Fig. 7.

Since the SIP invitation process is also based on a three-way handshake procedure (see Fig. 3), SIP servers become vulnerable to attacks similar to TCP-SYN.

In this case the attacker generates a spoofed INVITE and sends it to the proxy that allocates memory in order to administer the new session. The proxy firstly parses the message and then forwards it to the appropriate callee. If the callee exists and accepts the call, she generates an OK response and forwards it to the proxy. However, since the INVITE request is spoofed, the "from" header is not associated with a real user and thus no one will acknowledge this request (see Fig. 8), until a time-out is triggered. Consider now the case where, similar to a TCP/SYN attack, the malicious user generates numerous INVITE messages trying to turn the SIP proxy unavailable by exhausting its memory. This type of attack in SIP has been named "*SIP-INVITE SYN syndrome attack*" similar to the TCP-SYN attack.

An alternative flooding attack, although similar to the "*SIP-INVITE SYN syndrome attack*", is the generation of a spoofed message with irresolvable *uri* in the first line. In this case the proxy will also maintain the transaction up to 3 min. This kind of flooding attack has been already presented in Sisalem et al. (2006).

Table 1 – Description of the attack scenarios.

Scenario name	Scenario description
Scenario 1 (S1)	In this scenario the "*legal request generator*" generates (serially) requests (at a pace of 1 req/μs), while the corresponding responses are generated by the "*legal response generator*".
Scenario 2 (S2)	In this scenario the malicious user generates requests (at a pace of 1 req/10 μs) that are addressed to an innocent user who tries to respond to all of them.
Scenario 3 (S3)	In this scenario the malicious user generates requests (at a pace of 1 req/10 μs) that are addressed through the proxy to clients belonging to non-existing domains.

3.1.3. *INVITE-reflection syndrome attack*

A variation of the TCP-SYN attack was launched on 11th January 2002. In this attack innocent TCP servers were utilized as amplifiers (known as reflectors) to constitute a specific server out of order. Specifically, the attacker generates numerous TCP-SYN requests including spoofed IP addresses (victim's address). Consequently, the innocent servers generate a SYN/ACK forwarding it to the victim (see Fig. 9). This type of attack is known as distributed reflection DoS (DrDoS) (Gibson, 2002). Due to the fact that there is a vast amount of traffic, such an attack normally consumes all available bandwidth or exhausts the resources of the server.

An analogous attack can be launched against a SIP based service. Specifically, the attacker crafts spoofed INVITE messages and sends them to the innocent SIP servers who forward them to a specific target (see Fig. 10), trying to consume the memory of the SIP server by maintaining a large number of INVITE transactions. Alternatively, the SIP servers could be replaced by innocent User Agents that respond to the spoofed INVITE requests and forward the responses to the target, in a way similar to the TCP-SYN reflection attack. This type of attack in the SIP realm has been named "*INVITE-reflection syndrome attack*".

This specific attack can take various forms in SIP, depending on the spoofed SIP message. For instance an alternative of the "*SIP-INVITE forwarding*" is to add in the *Via* header the address of the victim and thus having the reflectors to forward the responses to the victim. Another way to pass the attack traffic through a specific SIP server is to embed the Record-Route header in the SIP spoofed message.

3.2. *Flooding attacks against end-users*

Flooding attacks against end-users are very similar to the ones against proxy servers. The main difference is that in all spoofed INVITE messages the request "*uri*" is the same and also the transmission rate of the messages is much lower compared to the flooding attacks against proxy servers since the processing capabilities of SIP clients are usually limited. Consequently when the number of requests exceed the number that the end-user can handle, she will not be able to respond and a DoS, for the specific end-user, will be caused. In such cases the proxy server is used as an amplifier. Fig. 11 illustrates a scenario where the malicious user sends, in

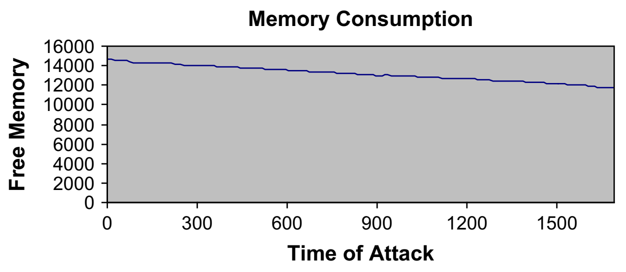


Fig. 13 – Memory consumption for Scenario 1.

random bursts, bogus INVITE messages, trying to cause a DoS to a specific legitimate user.

4. Launching flooding attacks in SIP realms

In order to simulate and explore the consequences of the flooding attacks presented in Section 3, we have employed the environment presented in Fig. 12.

Specifically, we have developed a SIP traffic generator consisting of two parts: (a) the request generator and (b) the corresponding response generator. The malicious SIP generator is "responsible" to create bogus SIP requests targeting either the SIP based Server or an innocent user depending on the type of the attack. For the innocent user and the SIP server we have utilized the open source SIP products (KPhone) and SIP Express Router (SER) accordingly. The server machine uses a Pentium 4 processor clocked at 2.5 GHz and 256 MB of RAM while the local network is an Ethernet at 100 Mbps. In all scenarios SER operates in stateful mode using the default parameters to handle the corresponding transactions.

To identify the consequences and mainly the memory consumption of flooding attacks in SIP, we have deployed the scenarios described in Table 1. The duration of each scenario was approximately 30 min. Figs. 13–15 present the memory consumption profiles.

It is noticed that the memory consumption rate of the SIP server when the system is under high rate of normal traffic (i.e. Scenario 1) is low, as clearly illustrated in Fig. 13. It is stressed that in this scenario the memory consumption is mainly caused by the fact that the established sessions between the "requestor" and the "responder" are not terminated but, instead, they remain active in the SIP server throughout the test in order to cause more stress to the SIP proxy.

On the contrary, when the system is under attack (i.e. Scenario 2) the memory can be exhausted rapidly (Fig. 14). In this case the user "co-operates" in the attack innocently. The innocent user has previously registered with the provided service and thus the proxy creates a new transaction and allocates the appropriate resources (mainly memory) for any new incoming request (invitation) in order to handle it and forwards the message to the user. The memory consumption is rapid due to the fact that the proxy keeps allocating memory for the new transactions without releasing it until a time-out is triggered, or a final response from the user described in the request is received. However, because the user is spoofed and thus does not exist, the memory is released only after a time-out is triggered (see Section 3.1.2 for more details on this type of attack). It is clear that if such an attack is launched in a distributed manner against a SIP proxy, it would cause limited access to the provided service or, in the worst case, it would exhaust the memory of the server causing a DoS.

Moreover, the attack traffic forwarded by the SIP proxy to the innocent user (see Fig. 11) results to a DoS for the user after only a few seconds. The end-user will not be able to react to any incoming request until its terminal is restarted. Nevertheless the innocent user could be replaced by a malicious one who would respond to the incoming requests in an attempt to cause a DoS to the proxy.

In Scenario 3 an alternative flooding attack is launched against the proxy. Specifically the attacker consumes system resources due to the fact that the SIP server generates DNS requests, in order to locate the requested resources, that in reality are irresolvable addresses. Consequently, these DNS requests block the SIP proxy server until a DNS response is received. A scenario demonstrating similar attacks has been also presented in Sisalem et al. (2006).

Last but not least, the INVITE flooding attacks, as illustrated by Scenarios 2 and 3, try to consume system's memory in order to cause DoS even though the system is under normal traffic.

It is thus necessary to detect such flooding attacks as soon as possible, in order to avoid their consequences and also to protect the end-users' terminals from DoS.

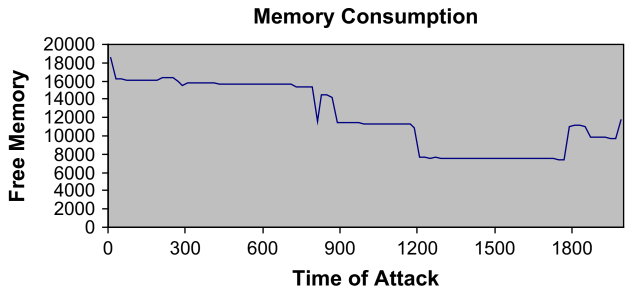


Fig. 15 – Memory consumption for Scenario 3.

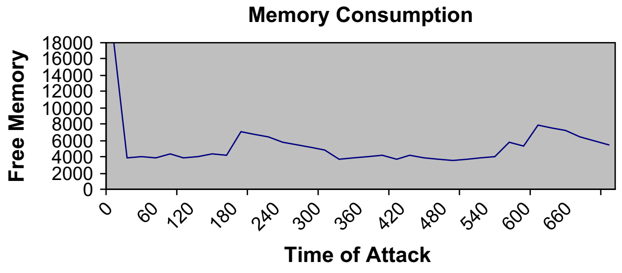


Fig. 14 – Memory consumption for Scenario 2.

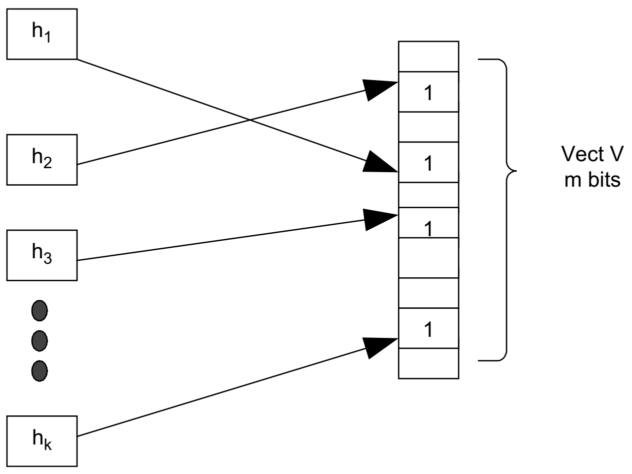


Fig. 16 – The original bloom filter.

5. A bloom filter protection method in SIP systems

5.1. An overview of the bloom filter

The bloom filter was introduced in Bloom (1970), proposing a space efficient data structure to store a specific set of elements and test the existence or not of a particular member of the set in the stored data structure by accepting allowable falsely identified members (false positives) of the given set. Specifically, a set A of elements $\{a_1, a_2, \ldots, a_n\}$ can be stored in a vector V of m bits. All bits of the vector V are initially set to zero. To each item of the set, k hash functions are applied. The hash function results are used as an index to the filter while the value of the corresponding entry is set to 1. It should be noted that the probability of a false positive depends on the parameters n, m and k. The above procedure is depicted in Fig. 16.

In the case that one would like to check the existence of a specific element (e.g. a_3) in the filter, she should apply the k hash functions to the a_3 element, reproducing the indexes to the specific filter entries. If any of these filter entries stores a 0 value, then a_3 is not part of the set stored in the filter. A detailed analysis of the bloom filter can be found in Bloom (1970) and Fan et al. (2000). An extended version of the bloom filter is proposed in Fan et al. (2000), where the vector V of m bits is replaced by a vector V' of m counters. In this version of the filter instead of modifying a single bit from 0 to 1, like in the original one, the corresponding counter is increased or decreased depending on filter's use. The specific implementation of the bloom filter has been employed for the detection or/and prevention of flooding attacks that launched against TCP servers as presented in Xiao et al. (2006).

5.2. The monitor system

The proposed detection mechanism utilizes bloom filters with counters. Specifically, in order to monitor, and consequently identify, flooding attacks against SIP proxies and end-users, a two-part bloom based monitor has been employed. Actually, the monitor's main task is to record the state (open, in progress, established) of any incoming session, information that will be utilized during the detection phase. Furthermore, it should be stressed that a single tracking memory solution can only be used as an indication of an attack, whereas the proposed monitor's information can be used, among others, for identifying malicious messages. On top of that the bloom filter is considered a cost effective and efficient method for recording–logging large amount of information in "compact" data stores.

The first part logs–monitors all new incoming requests, while the second one logs–monitors the requests that have been directed to a specific end-user. The first part of the monitoring system features three distinct bloom filters in order to keep track of (a) the INVITE requests, (b) the corresponding responses and (c) the final ACKs. The key input to the hash functions is the concatenation of the data included in the *call_id* and the *from* header (these headers can identify a session uniquely). Fig. 17 illustrates the proposed monitor

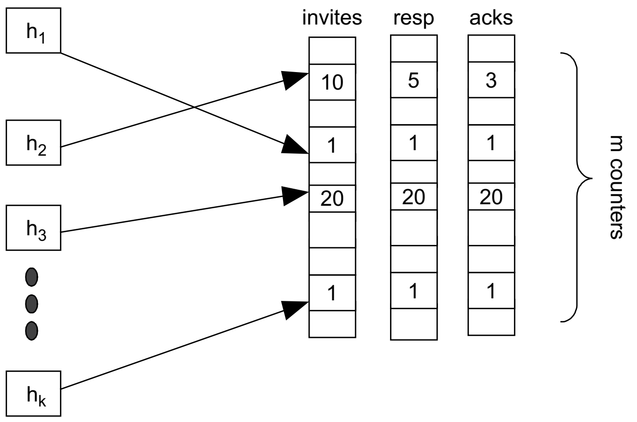


Fig. 17 – The first part of the SIP based monitor.

whereas Fig. 18 presents the proposed logging algorithm only as far as the first part is concerned.

It is important to stress that the employment of the three bloom filters allows the detection of flooding attacks using either SIP responses or final ACKs' messages.

The second part of the monitor keeps track of the sessions directed to a specific end-user. The main differences with the first part of the monitor are

1. the key input to the hash functions is the "*to*" header of the incoming request.
2. Only the "INVITE" segment (see Fig. 17) is employed, since the other segments are not required.

The monitoring algorithm of this part of the monitor is that for any new INVITE message the corresponding entries of the monitor are increased by one.

5.3. The detection method

In general, a flooding attack is associated with numerous incomplete sessions. On the contrary, in three-way handshake protocols, like SIP, for any valid session there is a unique one-to-one mapping among INVITEs–responses–ACKs. Therefore every INVITE message should be matched with one and only one response and acknowledgement. As illustrated in Fig. 19 the responses and ACKs should follow in time the INVITE.

```
For each incoming_message check the type
If type is request
  Check the method
    If the method is INVITE
      update the invite_bloom_filter
     Else if the medthod is ACK
      update the ack_bloom_filter
If type is response and is final response
   update the response_bloom_filter
```

Fig. 18 – General monitoring algorithm.

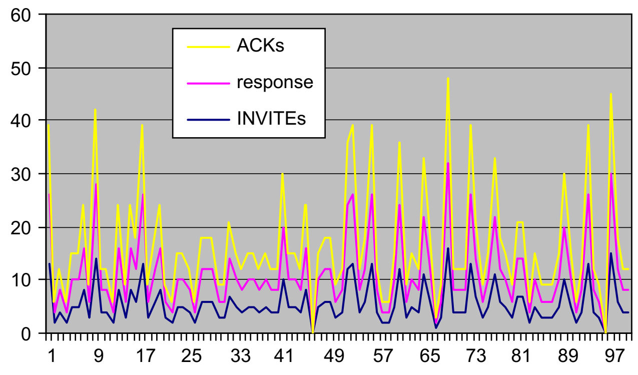


Fig. 19 – One-to-one mapping for SIP session.

The proposed detection mechanisms should also take into account (a) the average network delay (Nd) and (b) the average user response time (URT). Recalling that each INVITE message should be matched with the corresponding response and ACK, with a one-to-one mapping, we introduce the term *session distance* defined by the following metric:

$$\text{dist} = \text{Num of } \textit{INVITEs} - 0.5 * (\text{Num of } \textit{OK} + \text{Num of } \textit{ACK})$$

Clearly, in a well "behaved" environment the *session distance* metric for any established session is equal to zero. Based on the proposed mechanism, the existence of a distributed flooding attack can be identified by calculating, at specific intervals, the *session distance* metric for all the entries of the filter. The frequency of the checks, corresponding to a time interval T_{DDos}, depends on the SIP based system capabilities and resources.

The threshold for DDoS can be specified by "observing" the average value of the *session distance* metric during a specific period of system's operation (illustrated in Fig. 20). This can be considered as the "training period" of the DDoS module.

For instance, consider a case where the administrator of a particular realm observes the SIP traffic during peak hour and finds out that the average value of the *session distance* is T_{sd1}, and that the corresponding network parameters are Nd_1 and URT_1. The appropriate threshold value that should be adopted for identifying a DDoS attack is calculated through the following formula:

$$T_{alarm} = T_{sd1} + Nd_1 + URT_1 + \delta \quad (1)$$

where δ is a parameter reflecting the specific SIP system's capabilities and resources.

A DDoS alarm is triggered if the observed traffic exceeds T_{alarm}.

The first part of the monitor can be also utilized for detecting single-source flooding attacks and particularly cases where the attacker uses exactly the same message to launch a flooding attack against a proxy or a specific end-user. In this case a specific entry of the INVITE segment of the first part of the monitor keeps increasing, causing the increase, at the same pace, of the corresponding *session distance* values which although do not exceed the T_{alarm} value. A second threshold value, $T_{single1}$, is therefore necessary in order to detect flooding attacks that utilize the same message against a specific end-user. Consequently, if a computed *session distance*$_i$ value exceeds the threshold $T_{single1}$ an alert is triggered and the message is discarded.

However, if the attacker utilizes different SIP messages in order to launch flooding attack against a specific end-user, he may evade the detection mechanism since the incoming INVITEs will be distributed uniformly in the corresponding part of the monitor and thus the calculated *session distance*$_i$ will remain under the threshold $T_{single1}$. Nevertheless, the requested '*uri*' and the corresponding '*TO*' header will be the same as the attacker targets against a specific end-user. Thus, in this case, the utilization of the second part of the monitor is necessary. Each entry of this part of the monitor corresponds to the incoming request targeting to a specific user. If any of these entries exceeds the threshold $T_{single2}$, an alarm is triggered. This part of the monitor is checked every T_1 seconds, where T_1 is defined in RFC 3261 (Rosenberg et al., 2002) as the retransmission time of the INVITE request until it receives a response. The maximum number of retransmission is eight (8) while the typical retransmission delay is as shown in Fig. 21.

Thus the threshold $T_{single2}$ should not exceed the number of 8 retransmissions during 32 s.

The proposed detection mechanism can trigger specific administrative tasks, according to the security policy that has been employed, for mitigating the attack. For example, a service provider namely 'SIP-A' may employ a policy according to which when an incoming connection is recorded by the proposed monitor and the corresponding threshold is exceeded, it means that this specific session has already been "recorded" (by the mechanism) and thus it should be dropped. A different service provider namely 'SIP-B' may employ a different security policy according to which as long as the computed *session distance* is above the corresponding threshold value, all incoming messages are dropped.

5.4. *Evaluation*

Fig. 22 illustrates the network architecture utilized for the evaluation of the proposed scheme. The evaluation is done in terms of the overheads introduced and the effectiveness of the scheme. Specifically, the architecture consists of (a) a SIP server that implements the proposed mechanism (see Section

```
For all elements in the monitor do
    session_distance_i=num_of_invite_i-0,5*(num_of_resp_i+ num_of_ack_i)
    threhsold_value=+session_distance_i
```

Fig. 20 – DDoS threshold estimation procedure.

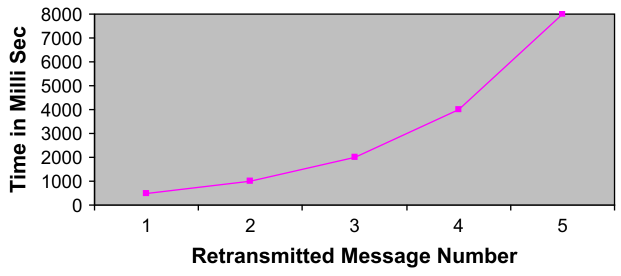


Fig. 21 – SIP-INVITE transactional mode.

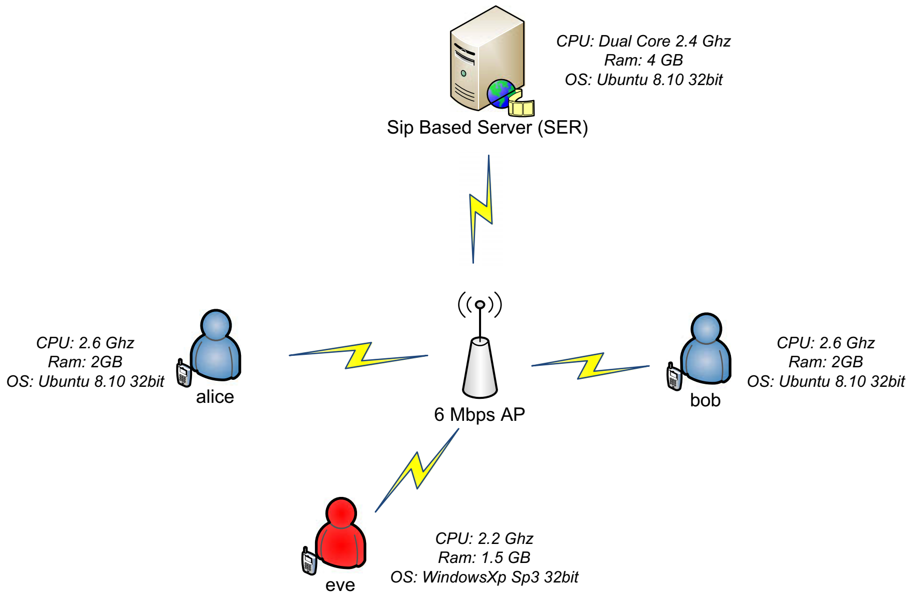


Fig. 22 – Test bed network architecture.

5.3) as an external module, using three hash functions for the log system; the well known open source SIP Express Router (SER) has been used (b) *a Legal Call Generator* (bob) and the corresponding *Legal Call Responder* (alice) and (c) *a Malicious SIP Call Generator* (eve). The particular technical characteristics of each of the components are also depicted in Fig. 22.

For the scope of the evaluation different scenarios have been implemented as presented in Table 2.

As far as the measurement of the detection time and of the overheads introduced, Figs. 23–25 depict the results for Scenarios 1–3 correspondingly, while Table 3 presents the statistical characteristics for each of them. At this point it should be stressed that in all scenarios the average detection time is under 40 μs. As illustrated in Fig. 26 the probability of having a detection time (for all scenarios) of less than 35 μs is near to 0.95.

Table 2 – Brief description of the employed scenarios.

Scenario name	Scenario description
Scenario 1 (S1)	In this scenario there is only legal traffic. Specifically the "*legal request generator*" generates requests (at a pace of 11 req/s), while the "*legal response generator*" generates the corresponding responses.
Scenario 2 (S2)	In this scenario the malicious user generates requests (at a pace of 150 req/10 s) that are addressed to a(n) (specific) innocent user (flooding an end-user; see also section III.B) who tries to respond to all of them with the existence of background traffic 11/cps
Scenario 3 (S3)	In this scenario the malicious user generates requests (at a pace of 1 req/10 μs) that are addressed through the proxy to various clients belonging to non-existing domains with the existence of background traffic 11/cps

Moreover, in order to identify the CPU load-overhead introduced by the proposed scheme we have measured the CPU load for all the scenarios (S1 to S3) with and without the proposed scheme; the results are presented in Fig. 27. Specifically, when the system is not under attack (S1) the CPU load is the same for both configurations. On the other hand, when the system is under attack (S2 and S3) the CPU load increases approximately 2–3 times as compared to a system that does not implement the proposed solution. This increase was expected since:

(a) in the case of an attack the system process more traffic including "costly" operations like hashing, file opening and writing; note that these operations are introduced by the proposed scheme and
(b) for the purposes of the tests, in Scenarios 2 and 3 whenever an attack is identified, instead of simply generating an alert, we re-initialize the system (to continue the test).

Considering the aforementioned results as well as that bloom filters can be implemented in hardware and thus minimize further the overheads introduced (Fiedler et al., 2007), it can be deduced that the detection time/overhead introduced by the proposed mechanism is negligible.

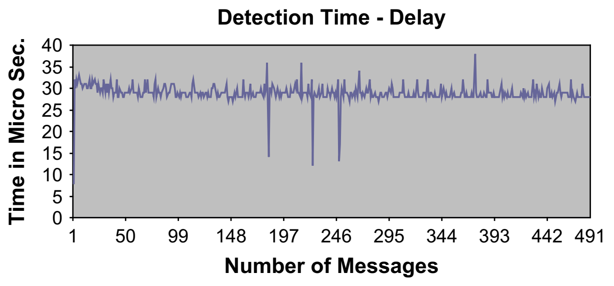


Fig. 23 – A sample for detection time-delay for Scenario 1.

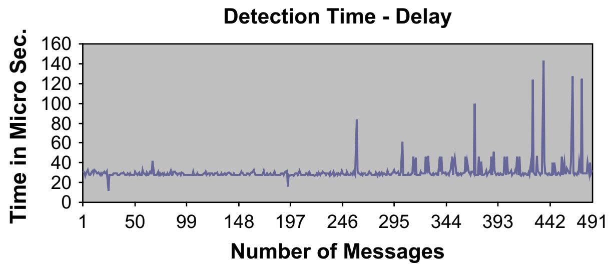


Fig. 24 – A sample for detection time-delay for Scenario 2.

Table 3 – Statistical attributes for Scenarios 1–3 with 95% confidence interval.

Scenario	Max.	Min.	Avg.
S1	29.13	29.00	29.06
S2	33.99	33.50	33.74
S3	29.03	28.90	28.96

Considering the effectiveness of the proposed scheme, as happens with most detection methods, it depends on the rate of false alarms (negatives and/or positives). In order to evaluate the effectiveness the corresponding thresholds (T_{alarm}, T_{single1}, T_{single2}) were defined according to the ''procedure'' presented in Section 4.3. Particularly, in order to train the proposed scheme (thresholds definition) we run *Scenario 1* (normal traffic) for 1 h. Note that if a different VoIP provider employed the proposed mechanism, the training phase should be repeated in order to take into account the different traffic characteristics (e.g. peak hour) between different providers. Using the DDoS threshold estimation procedure (see Fig. 20) and *formula 1* the threshold T_{alarm} for our system was estimated to

$$T_{\text{alarm}} = T_{\text{sd1}} + \text{Nd}_1 + \text{URT}_1 + \delta, \quad \text{where } T_{\text{sd1}} = 150, \ \text{Nd}_1 + \text{URT}_1 = 30 \text{ and } \delta = 0$$

while the values of T_{single1} and T_{single2} were set to 6 and 8 respectively.

It is stressed that no false alarms were triggered by S1, S2 and S3. However, in cases where the thresholds have not been set to the appropriate values, the proposed scheme generated false alarms, either negative or positive. For example, if for some specific reason (e.g. *mother's day*) the normal traffic exceeds the thresholds, the proposed scheme would generate false positive alarms. Nevertheless, in such cases the triggered (false) alarms could have a ''positive effect'' as they provide a warning about the excessive use of system resources.

6. Alternative countermeasures and remedies

SIP based VoIP systems employed in Internet are vulnerable to various kind of threats and attacks as already identified in Endler et al. (2005), Sisalem et al. (2005), Geneiatakis et al. (2006), while Ming et al. (2008) presents the side effects of flooding attacks in SER, validating the results presented in Section 4. On the other hand in Sisalem et al. (2006); Geneiatakis et al. (2007), Bagchi et al. (2004), Cao and Jennings (2006), Geneiatakis and Lambrinoudakis (2007) and Niccolini et al. (2006) address solutions for irresolvable DNS addresses, malformed messages and signaling attacks.

However, to the best of our knowledge, the research work on the identification of flooding attacks is still limited (Chen, 2006; Sengar et al., 2006; Reynolds and Ghosal, 2003; Ormazabal et al., 2008; Ehlert et al., 2008; Bouzida and Mangin, 2008). Particularly, paper (Reynolds and Ghosal, 2003) presents an application-layer attack sensor, based on the cumulative sum method, and the correlation between INVITE and OK messages in order to detect flooding attacks. Another interesting approach similar to Reynolds and Ghosal (2003) is described in Sengar et al. (2006). In this case the method is based on the correlation of INVITE, OK and BYE (or CANCEL), and the *Hellinger* distance among them. It is important to stress that the proposed mechanism differs from the solution presented in Sengar et al. (2006) in the following aspects: (a) the monitoring/recording of the incoming traffic is performed through an efficient and cost effective mechanism based on bloom filters and (b) the introduction of the session distance metric that focuses on SIP's session specific characteristics, which is less computational demanding as compared to the *hellinger* distance utilized in Sengar et al. (2006). Moreover,

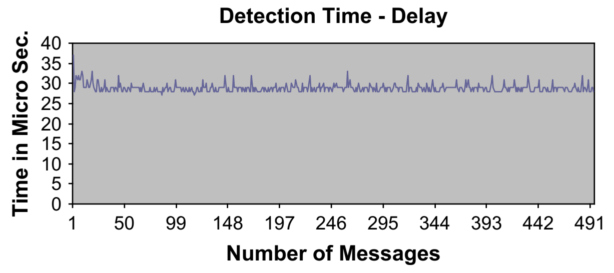


Fig. 25 – A sample for detection time-delay for Scenario 3.

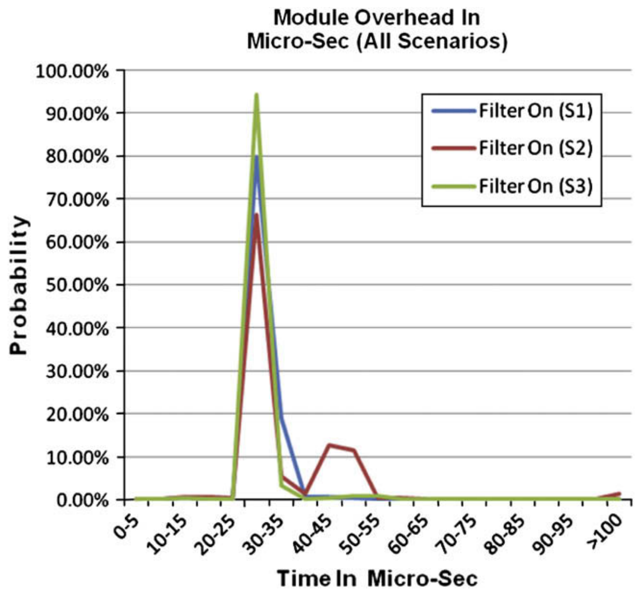


Fig. 26 – Probability density function.

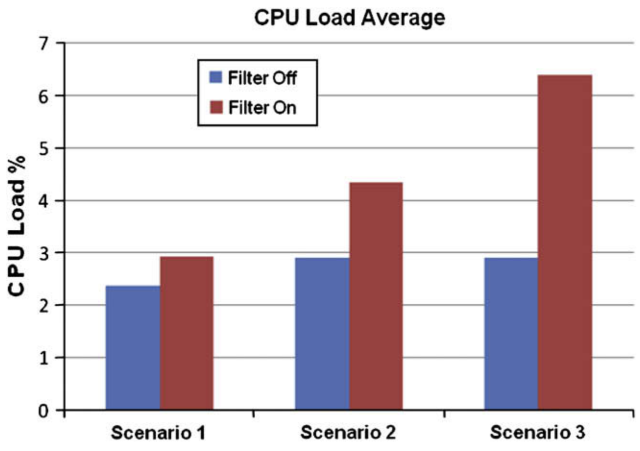


Fig. 27 – CPU Load for Scenarios 1–3.

Bouzida and Mangin (2008) introduce an attribute based system for identifying abnormal traffic, whereas in Chen (2006) and Ehlert et al. (2008) a different approach, based on a modified version of the original finite-state machine for SIP transactions, is followed. A slight variation of that solution is proposed in Ding and Su (2007) where instead of utilizing finite-states' machines for modeling the normal behavior of a SIP agent, it suggests the usage of colored petri networks. Finally Ormazabal et al. (2008) and Fiedler et al. (2007) focus on the design of the appropriate scalable architecture for protecting SIP based services against flooding attacks.

On top of that, one might assume that an authentication and authorization mechanism could protect the VoIP system against flooding attacks. It is true that the employment of such a mechanism could deter users from acting maliciously. In the context of SIP several authentication–authorization mechanisms have been proposed (Yang et al., 2005; Wu et al., 2009; Bremler-Barr et al., 2006; Chang et al., 2005; Mazurczyk and Kotulski, 2006; Peterson and Jennings, 2006; Tschofenig et al., 2006). However, it is highly unlikely that an authentication–authorization mechanism can be utilized for detecting–identifying and afterwards preventing any resource consumption–flooding attack. Furthermore, an authentication mechanism cannot provide protection against flooding attacks launched by insiders. On top of that there are various techniques (Geneiatakis et al., 2006) to bypass the authentication mechanism. For these reasons, the employment of the appropriate detection mechanism (as suggested in this work) should be considered mandatory.

7. Conclusion and future work

The emergence of SIP based VoIP services does not only gain the interest of service providers but also that of the attackers. It is beyond doubt that an attacker will try to expose and finally exploit any possible vulnerability in SIP systems as well as in any VoIP subsystem, aiming to harm their availability and trustworthiness. Various types of attacks against those sensitive real-time systems have been already reported, stemming mainly from VoIP open nature inherited by the Internet.

The paper presents flooding attacks against SIP based VoIP systems and correlates them to the well known TCP flooding attacks that are known in Internet services. A monitor, based on bloom filters, combined with a new metric, named ‘‘*session distance*’’, are introduced as a detection mechanism against such attacks. The proposed system was evaluated through different scenarios. The results have demonstrated not only that the detection time is negligible, but also that the rate of false alarms is minimal provided that the corresponding thresholds have been set correctly.

Although flooding attacks against VoIP systems have not been recorded yet, it is highly-likely that they will occur very soon. Thus the development of such detection mechanisms should be considered mandatory in order to increase the robustness of SIP based VoIP systems. Similar flooding attacks can be also launched against other signaling protocol like H.323 MCGP etc. Currently we are investigating extensions of the proposed mechanism in order to cover such cases and provide a cross platform identification method for flooding attacks.

REFERENCES

Gligor V. A note on denial-of-service in operating systems. IEEE Transactions on Software Engineering 1984;10:320–4.

Ping of Death, http://insecure.org/sploits/ping-o-death.html.

Gibson S. Distributed reflection denial of service Gibson Research Corp. Tech. Rep., Available from: http://grc.com/dos/drdos.htm; February 2002.

Carl G, Kesidis G, Brooks R, Rai S. Denial-of-service attack-detection techniques. IEEE Internet Computing 2006:82–9.

Peng T, Leckie C, Ramamohanarao K. Survey of network-based defense mechanisms countering the DoS and DDoS problems. ACM Computing Surveys 2007;39.

Mirkovic J, Dietrich S, Dittrich D, Reiher P. Internet denial of service: attack and defense mechanisms (Radia Perlman Computer Networking and Security). Upper Saddle River, NJ, USA: Prentice Hall PTR; 2004.

SIP swiss army knife (sipsak), http://sipsak.org.

Wieser C, Laakso M, Schulzrinne H. Security testing of SIP implementations. Available from: http://compose.labri.fr/documentation/sip/Documentation/Papers/Security/Papers/462.pdf; 2003.

Endler D, Ghosal D, Jafari R, Karlcut A, Kolenko M, Nguyen N, et al. VoIP security and privacy threat taxonomy public release 1.0. Voice over IP Security Alliance (VOIPSA) 2005;24.

Sisalem D, Ehlert S, Geneiatakis D, Kambourakis G, Dagiuklas T, Markl J, et al. Towards a secure and reliable VoIP infrastructure. Technical Report D2; 2005.

Geneiatakis D, Dagiuklas T, Kambourakis G, Lambrinoudakis C, Gritzalis S, Ehlert K, et al. Survey of security vulnerabilities in session initiation protocol. IEEE Communications Surveys and Tutorials 2006;8:68–81.

Ming L, Tao P, Leckie C. CPU-based DoS attacks against SIP servers. In: Network Operations and Management Symposium (NOMS 2008). IEEE; 2008. p. 41–8.

Rosenberg J, Schulzrinne H, Camarillo G, Johnston A, Peterson J, Sparks R, et al. SIP: session initiation protocol, RFC 3261, Internet Engineering Task Force; 2002.

Fielding R, Gettys J, Mogul J, Frystyk H, Masinter L, Leach P, et al. RFC2616: hypertext transfer protocol-HTTP/1.1. United States: RFC Editor; 1999.

Postel J. RFC 793: transmission control protocol; September 1981.
Center C. TCP SYN flooding and IP spoofing attacks, CERT Advisory CA-1996-21; September 1996. p. 1996–2021.
Sisalem D, Kuthan J, Ehlert S. Denial of service attacks targeting a SIP VoIP infrastructure: attack scenarios and prevention mechanisms. IEEE Network 2006;20:26–31.
Bloom B. Space/time trade-offs in hash coding with allowable errors. Communications of the ACM 1970;13:422–6.
Fan L, Cao P, Almeida J, Broder A. Summary cache: a scalable wide-area web cache sharing protocol. IEEE/ACM Transactions on Networking (TON) 2000;8:281–93.
Geneiatakis D, Kambourakis G, Lambrinoudakis C, Dagiuklas T, Gritzalis S. A framework for protecting a SIP-based infrastructure against malformed message attacks. Computer Networks 2007;51:2580–93.
Bagchi S, Wu Y, Garg S, Singh N, Tsai T. SCIDIVE: a stateful and cross protocol intrusion detection architecture for voice-over-IP environments. At IEEE. Dependable Systems and Networks (DSN 2004) 2004:433–42.
Cao F, Jennings C. Providing response identity and authentication in IP telephony. In: Proceedings of the first international conference on availability, reliability and security. IEEE Computer Society; 2006.
Geneiatakis D, Lambrinoudakis C. A lightweight protection mechanism against signaling attacks in a SIP-based VoIP environment. Telecommunication Systems 2007;36:153–9.
Niccolini S, Garroppo R, Giordano S, Risi G, Ventura S, Ltd N, et al. SIP intrusion detection and prevention: recommendations and prototype implementation. In: 1st IEEE workshop on VoIP management and security; 2006. p. 47–52.
Chen E. Detecting DoS attacks on SIP systems. In: 1st IEEE workshop on VoIP management and security; 2006. p. 53–8.
Sengar H, Wang H, Wijesekera D, Jajodia S. Fast detection of denial of service attacks on IP telephony. In: Proceedings of the 14th IEEE international workshop on quality of service (IWQoS 2006); 2006. p. 199–208.
Reynolds B, Ghosal D. Secure IP telephony using multi-layered protection. In: Proceedings of the network and distributed system security symposium (NDSS 2003); February 2003.
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; 2008. p. 107–32.
Ehlert S, Wang C, Magedanz T, Sisalem D. Specification-based denial-of-service detection for SIP voice-over-IP networks. In: Proceedings of the third international conference on internet monitoring and protection. IEEE Computer Society; 2008.
Bouzida Y, Mangin C. A framework for detecting anomalies in VoIP networks. In: Third international conference on availability, reliability and security, ARES 08; 2008. p. 204–11.
Franks J, Hallam-Baker P, Hostetler J, Lawrence S, Leach P, Luotonen A, et al. RFC2617: HTTP authentication: basic and digest access authentication. Internet RFCs 1999.
Fiedler J, Kupka T, Ehlert S, Magedanz T, Sisalem D. VoIP defender: highly scalable SIP-based security architecture. In: Proceedings of the 1st international conference on principles, systems and applications of IP telecommunications; 2007. p. 11–7.
Ding Y, Su G. Intrusion detection system for signal based SIP attacks through timed HCPN. In: Proceedings of the second international conference on availability, reliability and security, 2007. ARES 2007; 2007. p. 190–7.
KPhone. A voice over Internet phone, http://www.wirlab.net/kphone.
SIP Express Router, http://www.iptel.org/ser.
SIP Express Router documentation, http://www.iptel.org/doc.
Xiao B, Chen W, He Y. A novel approach to detecting DDoS attacks at an early stage. Journal on Supercomputing 2006: 235–48.
Yang C, Wang R, Liu W. Secure authentication scheme for session initiation protocol. Computers & Security 2005;24:381–6.
Wu L, Zhang Y, Wang F. A new provably secure authentication and key agreement protocol for SIP using ECC. Computer Standards & Interfaces 2009;31:286–91.
Bremler-Barr A, Halachmi-Bekel R, Herzliya I, Kangasharju J. Unregister attacks in SIP. In: Proceedings of the 2nd IEEE workshop on secure network protocols; 2006. p. 32–7.
Chang C, Lu Y, Pang A, Kuo T. Design and implementation of SIP security. In: The proceedings of information networking convergence in broadband and mobile networking; February 2005.
Mazurczyk W, Kotulski Z. New VoIP traffic security scheme with digital watermarking. Lecture Notes in Computer Science 2006;4166:170.
Peterson J, Jennings C. Enhancements for authenticated identity management in the session initiation protocol (SIP), RFC 4474; August 2006.
Tschofenig H, Falk R, Peterson J, Hodges J, Sicker D, Polk J, et al. Using SAML to protect the session initiation protocol (SIP). IEEE Network 2006;20:14–7.

Dr. Dimitris Geneiatakis was born in Athens, Greece, in 1981. He received the five-year Diploma in Information and Communication Systems Engineering in 2003, and the M.Sc. in Security of Information and Communication Systems in 2005, and a Ph.D. in the field of Information and Communication Systems Security from the Department of Information and Communications Systems Engineering of the University of Aegean, Greece. He has participated in various national and international projects in the area of Information Systems Security. His current research interests are in the areas of security mechanisms in Internet Telephony, Smart Cards, Intrusion Detection Systems and Network Security. He is an author of several refereed papers in international scientific journals and conference proceedings. He has served on program and organizing committees of international conferences on Informatics and is a reviewer for several scientific journals. He is a member of the Technical Chamber of Greece.

Nikos Vrakas holds a B.Sc. degree in Mathematics and an M.Sc. in Security of Information and Communication Systems. His current research interest is in the areas of Critical Infrastructures Availability, Intrusion Detection Systems and Security Mechanisms in IP Multimedia Subsystem.

Assistant Professor **Costas Lambrinoudakis** (B.Sc., M.Sc., Ph.D.) was born in Greece in 1963. He holds a B.Sc. (Electrical and Electronic Engineering) degree from the University of Salford (UK), an M.Sc. (Control Systems) and a Ph.D. (Computer Science) degree from the University of London (UK). Currently he is an Assistant Professor at the Department of Information and Communication Systems Engineering, University of the Aegean, Greece and the Associate Director of the Laboratory of Information and Communication Systems Security (Info-Sec-Lab). He has been involved in several national and EU funded R&D projects in the areas of Information and Communication Systems Security. These research programs include SERENITY, e-

SENSE, SNOCER (FP6 SME-1), e-VOTE (IST), HERMES (Telematics), GESTALT (ACTS), VSAT Network for Telematics and Health Care (SfS NATO), VITAL-Home (ISIS), IRIS (IST), etc. His published scientific work includes six books on Information and Communication Technologies' topics, and more than sixty journal and national and international conference papers. The focus of these publications is on Information and Communication Systems Security. He has served on program and organizing committees of national and international conferences on Informatics and is a reviewer for several scientific journals. He is a member of the ACM and the IEEE.