Novel Protecting Mechanism for SIP-Based Infrastructure against Malformed Message Attacks: Performance Evaluation Study

Dimitris Geniatakis, Tasos Dagiuklas,
Costas Lambrinoudakis, George Kambourakis and Stefanos Gritzalis

InfoSec Lab
Dept. of Inform. & Comm. Syst. Eng., University of Aegean
83200, Karlovassi, GREECE

Abstract—This paper presents a novel mechanism to protect Session Initiation Protocol (SIP)-based infrastructure against malformed message attacks. The basic characteristics of this mechanism are the following: lightweight and easy to adapt to various SIP implementations. The proposed mechanism has been evaluated in terms of overhead processing. It is demonstrated that the employment of appropriate IDS against malformed impose minimum overhead in terms of events' processing

***Index Terms*—SIP attacks, Performance Evaluation**

I. Introduction

Is t is known that the Internet is susceptible to a plethora of attacks, and without a doubt, must be considered a hostile environment for every critical real time application like Internet Telephony. This means that attackers will try to discover and potentially exploit special vulnerabilities found either in signaling or voice transport. Until now, various researches [1], [2] have made significant efforts in identifying such security vulnerabilities that directly affect VoIP based infrastructures. For instance, VoIP services can be attacked in the usual way utilizing flooding techniques or more skillfully by employing specialized malformed packets.

Both protocol implementations and network applications are often not fully conformant with the underlying standards (e.g. RFCs) or they contain development errors in their source implementation code [3], [4]. Furthermore, standard protocol implementations focus on well formed messages and usually they do not consider any defense tactics against malformed messages. Once an attacker floods a SIP proxy with a number of malformed messages the victim is unable to process, and maybe discard them.

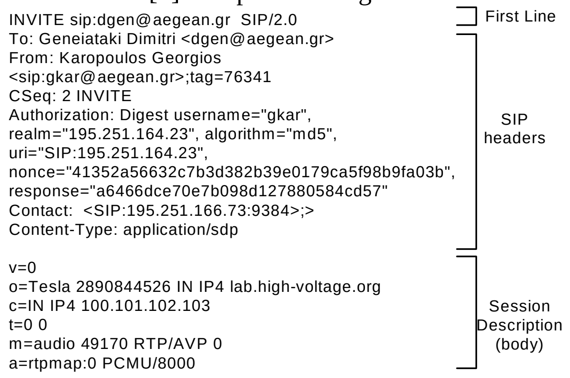
The term "malformed message" is referred to any kind of invalid or non-standard message in order to exploit and eventually either take advantage of any implementation gap or dysfunction might exist in the target system. Malformed messages are characterized as a high-level type of attacks associated with illegally formatted input. There are numerous types of malformed message [5], [6]. There is much interest on how these attacks can be extended to new and different types of Internet applications and services.

This paper proposes a novel mechanism to protect SIP-based subsystems (e.g,. SIP proxy) against malformed message attacks. The first part of the paper describes different variations of the malformed message attack within SIP subsystems and proposes a protection framework, consisting of prototyped attacks signatures. The second part of the paper presents the evaluation of this tool in terms of overhead processing. The remainder of the paper is organized as follows: Section II introduces the SIP malformed message attack. Section III provides our identification and prevention mechanism to protect SIP based networks against this class of attacks. Section IV evaluates the proposed solution in terms of the overhead that introduces in the corresponding subsystems. Finally, Section V concludes the paper providing some pointers to future work.

II. MALFORMED MESSAGES IN SIP

A. SIP Overview

SIP is an application-layer signaling protocol for creating, modifying, and terminating multimedia sessions between one or more participants [9]. SIP messages can be either a request or an acknowledgment to a corresponding request, consisting of the header fields and optionally a message body. The overall structure of a typical well formed SIP message based on RFC 3261 [9] is depicted in Figure 1.



```
INVITE sip:dgen@aegean.gr  SIP/2.0
To: Geneiataki Dimitri <dgen@aegean.gr>
From: Karopoulos Georgios
<sip:gkar@aegean.gr>;tag=76341
CSeq: 2 INVITE
Authorization: Digest username="gkar",
realm="195.251.164.23", algorithm="md5",
uri="SIP:195.251.164.23",
nonce="41352a56632c7b3d382b39e0179ca5f98b9fa03b",
response="a6466dce70e7b098d127880584cd57"
Contact:  <SIP:195.251.166.73:9384>;>
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
```

Figure 1: A well formed SIP-INVITE message

A SIP-based multimedia connection between two users is established whenever the caller (e.g. User A) sends a SIP INVITE message to the corresponding proxy, which in turn forwards it towards the User B (callee). The signaling flow procedure is depicted in Figure 2.

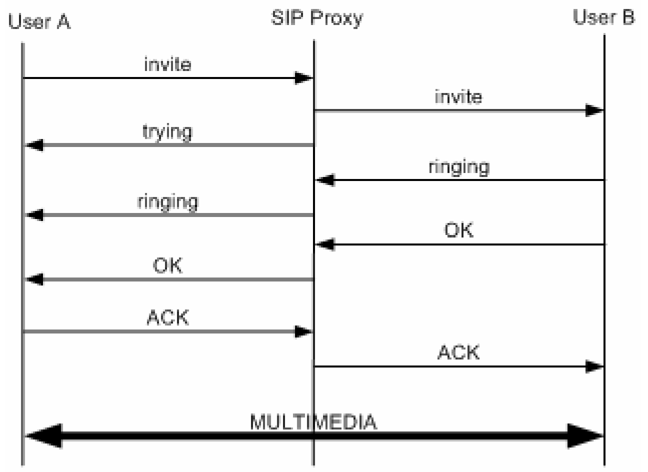


Figure 2: SIP connection establishment

Consequently, whenever a SIP request has been received from the corresponding SIP proxy and independently from the implementation. The following events take place when an incoming message is received by a SIP proxy:

The first operation that will be executed is to (initially) parse the SIP message. The parsing procedure is essential in order to represent the incoming request into an appropriate form. This form will be utilized in a later phase to construct the reply.

Figure 3 depicts the (initial) processing procedure which is executed by standard SIP proxies whenever they receive a request or response. Although some SIP proxies' implementations, may slightly vary the aforementioned procedure, the sequence described in steps 1 to 3 covers the general idea of the processing mechanism of SIP messages.

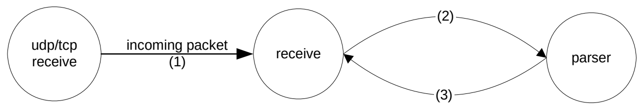


Figure 3: SIP connection establishment

B. SIP Malformed Messages

Generally, SIP parsers are developed to receive and process well-formed messages. By "well formed" we mean all SIP messages which conform to the RFC's 3261 syntax [9]. However, an attacker or even a poorly-implemented client is quite possible to send various types of distorted messages [11] in order to induce undesired situations such as DoS, Unstable operations and unauthorized access.

These problems are caused mainly because the parser in the SIP proxy is not able to handle and successfully process (e.g. drop) the received malformed messages. For example, during the session establishment phase (see Figure 2), an attacker could send various malformed message combinations in order to discover a security problem or flaw at the parser in the SIP proxy. As an example, instead of sending a well formed SIP INVITE as expected (see Figure 1), it sends the SIP INVITE message illustrated in Figure 4. However, this message is invalid and can not be generated by the standard SIP protocol syntax, due to the lack of a REQUEST-URI, which must always follow the SIP INVITE method [9]. The target of such a message is either a SIP proxy or the user's terminal (callee). The attacker possibly will not try only the SIP INVITE method but also the others as described in RFC 3261 [9] and any additional extension. More details for this kind of attack can be found in [11].

```
INVITE (null)
To: Geneiataki Dimitri <dgen@aegean.gr>
From: Karopoulos Georgios
<sip:gkar@aegean.gr>;tag=76341
CSeq: 2 INVITE
Authorization: Digest username="gkar",
realm="195.251.164.23", algorithm="md5",
uri="SIP:195.251.164.23",
nonce="41352a56632c7b3d382b39e0179ca5f98b9fa03b",
response="a6466dce70e7b098d127880584cd57"
Contact:  <SIP:195.251.166.73:9384>;>
Content-Type: application/sdp
```

SIP header

```
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
```

Session Description

Figure 4: Example of Malformed SIP INVITE message

III. PROPOSED MECHANISM TO PROTECT SIP-BASED INFRSTRUCTURE

A. Detecting Illegal SIP Messages

The availability of the VoIP subsystems can be reduced due to the fact that parsers in signalling servers like SIP proxies do not incorporate mechanisms to detect illegal SIP messages. The lack of any validation mechanism in the receiving process could be responsible for various security flaws. The employment of mechanisms to filter malicious input at the Internet application level has also been already investigated by some researchers [13]. Even state-of-the-art firewall technologies incorporate deep packet inspection methods [14].

Moreover, according to RFC 3261 [9] the utilization of underlying security mechanisms like SSL, IPsec, S/MIME and HTTP Digest can substantially restrict or prevent the origination of malformed messages. Furthermore, these mechanisms introduce additional traffic, and processing overhead to the corresponding SIP server. Nevertheless, these security schemes when utilized in SIP require the installation of an end-to-end or layered Public Key Infrastructure (PKI) beforehand. However, the biggest obstacle is the fact that the vast majority of vendors do not provide end-user devices that support security protocols like SSL [16].

Most importantly, all the aforementioned security protocols in some cases are proved to be ineffective. For example, as stated in [11], an attacker might utilize a SIP proxy from another realm to amplify the hazardous effects of the malformed messages that sends. Consequently although the SIP proxy may do not become crashed it will forward the malformed message towards other proxies in the path and finally towards the end-user trying to cause a DoS. In addition, these mechanisms do not provide any real security against (malevolent) insiders as it is well known that a lot of security incidents are originated by this group of users.

B. Detecting Mechanism for SIP Malformed Messages

No matter how strong the existing security mechanisms in VoIP services are, there is always the possibility for a malicious user to manage to by-pass them. The introduction of an appropriate detection and prevention module for malformed messages in the existing VoIP infrastructure must be considered as an important element to assure reliability and prevent DoS. Malformed message attacks can be described effectively through some specific static structure known as the attack's "signature".

The basic idea to build an identification mechanism for malformed messages stems from the SIP syntax as described in the RFC 3261 [9]. Any message which does not comply to the previous RFC can be characterized as malicious. Consequently, each of the proposed signatures is composed of two different parts. The first identifies malformed message. It is a general signature that can be applied to any SIP method. The second part specifies some optional rules that must be applied only for specific SIP methods and are defined by SIP's domain security policy. An example of this general signature is depicted in Figure 5.

```
SIP_METHOD SIP-URI | SIPS-URI MESSAGE HEADER+
[MESSAGE_BODY]

additionall rules
SIP_METHOD!=NULL
MESSAGE_HEADER!=NULL
size_of(SIP_METHOD)>%constant% e.g 50 bytes
size_of(MESSAGE_BODY)>%constant%
```

Figure 5: General Detection Signature

Notice, that the fist two lines makes mandatory that any SIP message must include the following:

- a SIP_METHOD, with a SIP or SIPS URI followed by the corresponding HEADERS.
- Both SIP_METHOD and the MESSAGE_HEADER must not be equal to NULL.
- It makes also clear that the MESSAGE_BODY is optional and its presence depends on the SIP_METHOD used

However there are certain circumstances of "well" structured malicious messages that can not be identified by this generally rule. For these instances special signatures must be formed for each distinct SIP-method. For example, according to the SIP standard syntax, SIP INVITEs must include at least one of some specific headers like Call-ID or Content-Type. Now, consider the case in which an incoming INVITE does not include any of these headers. This message must be characterized as malicious and must be discarded prior it is handled by the parser.

C. Enhancing SIP Subsystems Functionality

In order to protect SIP servers and users' terminals against malformed message attacks a pre-filtering module has been designed to discard all non well-formed SIP messages. Consequently, it is critical that this module must be inserted and operate prior any message goes to the parser. It is implied that the proposed solution can be also integrated into the system's core functionality. Then, when an incoming message matches any of the specified signatures is instantly identified as malformed and it is discarded. At the same time, as already noticed, the system appends a record with the discarded message to the corresponding file. When a specified threshold is violated (e.g. the system logs four or more malformed messages in one second) it activates an alarm to the operator's console. Figure 6 illustrates the modification that is required in any SIP proxy or user's terminal architecture. Notice, that this module can be also embedded inside a firewall that is able to "understand" SIP traffic. However, in this paper we concentrate on enhancing SIP servers' behavior without modifying other VoIP components or the general architecture of the SIP infrastructure.

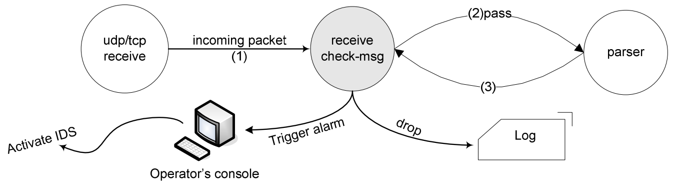


Figure 6: Enhancing SIP Proxy with Message checking

Hence, in order to avoid false alarms as well as the expressive overhead to develop a vast number of distinct signatures to cover all the possible combinations between methods and headers, rules have been used, based on regular expressions. The proposed schema is divided in three distinct inspection phases: First Line inspection, Header inspection and Specialized inspection

More specifically, general detection signature includes scanning procedures which relate to all the incoming SIP messages independently from the corresponding method. Therefore, all the incoming messages are validated both for first line correctness and (all) embedded headers appropriateness. Note, that the most common headers found in SIP messages are Via, From, To, and Content-Length.

The first line of any incoming message is examined against the existing malformed rules. If the first line conforms to the standard SIP syntax then scanning continues to the second stage, namely the Header inspection. When a header is found to be malicious, further processing of the message is paused, the "Check-Msg" "module" drops the message and records it into a "Bad-Transactions" file.

Furthermore, due to the different description syntax of SIP messages specialized inspection is required based on the method that has been identified in the first stage. As an example, INVITE and REGISTER methods do have some discrepancies. In particular, REGISTER does not include a message body contrariwise to INVITE that usually contains one. So a REGISTER which includes a message body must be characterized as malicious. However, we have to be very careful when determining such a restriction due to the relative freedom that exists of the SIP syntax. For this reason, some of these specialized controls must be determined from the local administrator of any different SIP realm.

Moreover, the Specialized inspection phase includes controls that utilize the combination of the corresponding method and header information. For example, when an INVITE message is received the CSEQ header must be in the following form: CSEQ: identification_number INVITE. In this context, a malevolent user may send an INVITE message in which the CSEQ header has the following syntax: CSEQ: identification_number REGISTER. Although this syntax, based on the RFCs, can be characterized as well formed, the parser must ignore it as includes a logical error (it contains a REGISTER statement instead of INVITE).

Furthermore, specialized inspection contain data validation to shield against SQL injection attacks. As a result, the SIP message syntax will be validated without any error but the authorization header may contain data that will possibly try to cause a modification in the local users' database.

IV. Performance Evaluation and Discussion

In order to evaluate the proposed SIP-Detection and Prevention mechanism, a SIP-based network testbed has been used. It consists of two terminals running SIP client software using the KPHONE [18] application and the appropriate SIP server using the SIP Express Router SER [10]. Furthermore, two different attack tools was employed; the first one was the SIPBOMBER [19] while the second one was a custom-made tool to create more sophisticated malformed messages. The SIPBOMBER tool utilizes tests contained in PROTOS test suite [8]. We also made the necessary modifications and additions to the SER [10] core engine to successfully exploit the signature-database and consequently identify malformed message attacks as described in Section III.C.

The most important comparative parameter in the processing overhead procedure of any incoming message induced from the enhanced SIP proxy core is the required time duration to process a well formed (normal) message. This metric will finally designate whether the proposed interweaved in the SIP's server IDS/IPS scheme has reasonable overheads. We must emphasize that all the processing times which are presented in the current section represent "worst cases" as the malicious messages we contracted, deliberately contained several malformed fields (errors). As a result, the embedded identification mechanism intentionally does not drop the message upon detecting the first malformed field but continues until all inspections have been completed and then drop the message if required. All the eight scenarios we developed to test the proposed mechanism, are described in the Table 1.

Scenario Number	Scenario Description
Scenario 1 (S1)	This scenario utilizes the SIPBOMBER tool to create malformed messages as described in PROTOS suite.
Scenario 2 (S2)	This scenario utilizes our custom-made tool to generate specific malformed messages that contain errors in one header only.
Scenario 3 (S3)	This scenario utilizes our custom-made tool to generate specific malformed messages that contain errors in the first line only.
Scenario 4 (S4)	This scenario utilizes KPHONE to generate well formed messages.
Scenario 5 (S5)	This scenario utilizes our custom-made tool to generate well formed messages.
Scenario 6 (S6)	This scenario utilizes our custom-made tool to generate malformed messages that contain errors in the following headers: From, To, Via, CSEQ.
Scenario 7 (S7)	Same as the previous scenario with the addition of SQL injection malicious code in the authorization header.
Scenario 8 (S8)	This scenario utilizes our custom-made tool to generate various well formed messages.

Note, that scenarios 1 to 4 leave out inspection procedures for authorization header and consequently for SQL code injection attacks. In addition, header inspections procedures include not only the normal header validation but also some specialized scanning, like CSEQ header syntax, logical controls (see sections III.C & D), the existence of multiple headers that must be unique e.g. FROM, TO, etc.

More specifically, Figures 7 & 8 illustrates the processing times and various statistical parameters for First line inspection, which remain to the average case under than 35 microseconds. It is obvious that all the plots seem to have similar distribution. This fact comes naturally as the only modifications between these scenarios were the different length of the first line and various changes in the form of the malformed messages we tested, trying to evade the detection mechanism. The moderately high standard deviation times, especially for scenario 1 can be explained due to the fact the SIPBOMBER tool and partially our-custom made application generate malformed packets of excessive length. In addition, some other services running on the SIP server might affect instantly the processing times producing peaks to the plots.

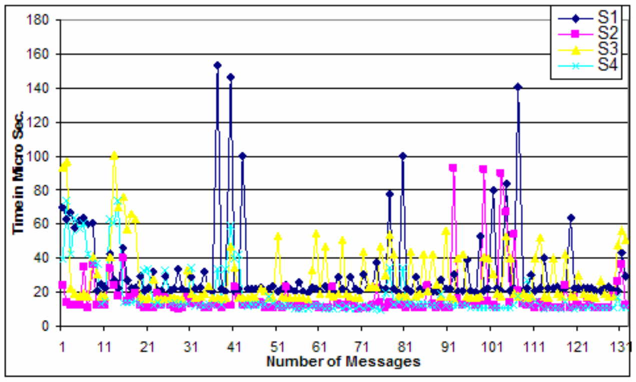


Figure 7: First Line inspection Time Overheads for Scenarios 1 to 4

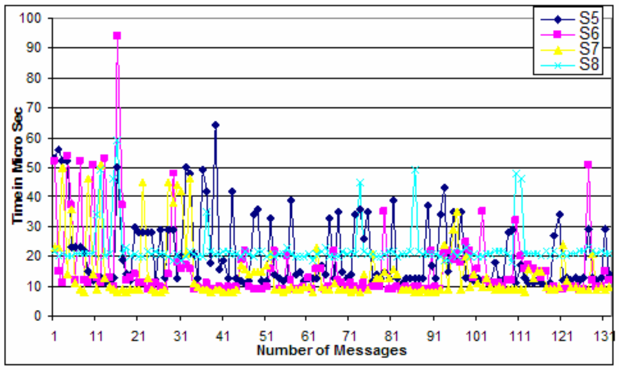


Figure 8. First Line inspection Time Overheads for Scenarios 5 to 8

All processing times remain to the average case under 120 microseconds except those for scenarios 6 & 7. The later scenarios induce an additional overhead to incoming message processing time as they were deliberately formed to contain more than one error. In particular, malformed messages for scenario 6 included four header errors, while scenario 7 deployed messages that had the same number of header errors and additionally one of them included SQL malicious code..

Beyond doubt, the maximum time of 120 microseconds that is being introduced by the proposed mechanism is insignificant. First and foremost, all processing times involving malformed messages are nearly the same with scenarios that employ well-formed messages (scenarios 4, 5 and 8). Another important comparative metric is the time that is required to establish a SIP connection which normally is more than one second. Moreover, the introduction of a new header that requires deep inspection like authorization checking in the SQL injection code case shows an additional time duration of about 80 microseconds. Further on, a potentially malformed message has statistically 50% probability to be discarded if an anomaly is detected in its first half, thus normalizing the overall processing time near the average as the time passes and the corresponding system is under attack (e.g. malformed message flooding). Whether or not, most of the time, systems are not under attack [20], so the anticipated scheme is expected to parse messages with an average time of under 150 microseconds as in scenarios 4, 5 & 8. For one more time, standard deviation parameter remains moderately high for most of the scenarios deployed. This fact is explained for the same reasons we illustrated previously.

Finally, to compare the overall reliability and performance of the SIP server that includes the proposed mechanism with a standard one, running SER, we launched a flooding attack with malformed messages against both of them. Standard SIP server crashed only after few seconds, in contrast to the enhanced SIP server that discarded all malicious traffic and continued to operate smoothly without any troubles.

V. CONCLUSIONS AND FUTURE WORK

In this paper SIP malformed message attacks have been analyzed focusing on SIP domains. We have presented attacks against different SIP subsystems, exploiting implementation “errors” sending malformed messages and launching an SQL injection attack correspondingly. Through experimentation, the proposed solution has been evaluated in terms of robustness and processing overhead, considering well-respected SIP products (server and client software). The derived times have demonstrated that the proposed solution is robust, flexible, feasible to implement and above all secure. Moreover, the proposed scheme and its associated signatures database can be easily applied or added up to other VoIP signaling protocols (e.g. H.323), firewalls or other open source IDS products.

ACKNOWLEDGEMENTS

This work has been performed in the framework of the IST-2004-005892 project SNOCER, (www.snocer.org), which is funded by the European Union.

REFERENCES

[1] VOIPSA, "VoIP Security and Privacy Threat Taxonomy" http://www.voipsa.org/Activities/taxonomy.php, October 2005

[2] Sisalem D., Ehlert S., Geneiatakis D., Kambourakis G., Dagiuklas T., Markl J.,Rokos M.,Botron O.,Rodriguez J., Liu J., Towards a Secure and Reliable VoIP, http://www.snocer.org, May 2005

[3] “Asterisk SIP Implementation Issue”, http://www.atstake.com/research/advisories/2003/a090403-1.txt, August 2003

[4] CERT® Advisory CA-2003-06, “Multiple vulnerabilities in implementations of the Session Initiation Protocol (SIP)”, http://www.cert.org/advisories/CA-2003-06.html February 2003.

[5] CERT-In Advisory CIAD-2003-09 “Buffer Overrun In RPC Interface Could Allow Code Execution and Denial of Service” August 2003.

[6] Fontana J., “Exchange Server 5.5 Bug Could Be Exploited for Attacks”, http://www.pcworld.com/resource/article/0,aid,33882,00.asp , November 2000.

[7] Paxson V., Auman M., Dawson S., Fenner W., Griner J., Heavens J., Labey K., Semke J., and B.Volt,. “Known TCP implementation problems”. RFC 2525, March 1999.

[8] Wieser C, Laakso M, Schulzrinne H , “Security testing of SIP implementations”, http://compose.labri.fr/documentation/sip/Documentation/Papers/Security/Papers/462.pdf, 2003.

[9] Rosenberg J., Schulzrinne H., Camarillo G., Johnston A., Peterson J., Spark R., Handley M., Schooler E., “Session Initiation Protocol”, RFC 3261, June 2002.

[10] “SIP Express Router”, http://www.iptel.org/ser

[11] Geneiatakis D., Kambourakis G., Dagiuklas T., Lambrinoudakis C. and Gritzalis S., "A Framework for Detecting Malformed Messages in SIP Networks", to appear in the Proceedings of 14th IEEE Workshop on Local and Metropolitan Area Networks (LANMAN), September 2005, Chania-Crete, Greece.

[12] Geneiatakis D., Kambourakis G., Lambrinoudakis C, Dagiuklas T, and Gritzalis S., “SIP Message Tampering: THE SQL code INJECTION attack”, to appear in the Proceedings of 13th International Conference on Software, Telecommunications and Computer Networks (SoftCOM 2005), September 2005, Split, Croatia

[13] Scott D. and Sharp R., “Abstracting Application-Level Web Security,” Proc. 11th Int’l World Wide Web Conf., ACM Press, New York, May 2002, pp. 396-407.

[14] Dharmapurikar S., Krishnamurthy P., Sproull T., and Lockwood J., “Deep Packet Inspection Using Parallel Bloom Filters.” In Proceedings 11th Symposium of High Performance Interconnects (HOTI’03), pages 44-71, 2003.

[15] Srisuresh P., Kuthan J., Rosenberg J., Molitor A. and Rayan A: “Middlebox Communication Architecture and framework”, IETF, RFC 3303, August 2002.

[16] Geneiatakis D., Kambourakis G., Dagiuklas T., Lambrinoudakis C. and Gritzalis S., "SIP Security Mechanisms: A state-of-the-art review", Proceedings of the Fifth International Network Conference (INC 2005), July 2005, Samos, Greece.

[17] Perl Compatible Regular Expressions http://www.pcre.org

[18] KPhone A Voice over Internet phone, http://www.wirlab.net/kphone/

[19] SIPBOMBER http://www.metalinkltd.com/downloads.php

[20] K. G. Anagnostakis, E. P. Markatos, S. Antonatos, and M. Polychronakis. E2xB: A domain–specific string matching algorithm for intrusion detection. Proceedings of the 18th IFIP International Information Security Conference (SEC2003), May 2003