Available online at www.sciencedirect.com
ScienceDirect
Computer Communications 30 (2007) 1367–1374
computer communications
www.elsevier.com/locate/comcom

An ontology description for SIP security flaws

Dimitris Geneiatakis *, 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

Received 17 June 2006; received in revised form 21 December 2006; accepted 29 December 2006
Available online 10 January 2007

Abstract

Voice over IP (VoIP) services based on the Session Initiation Protocol (SIP) gain ground as compared to other protocols like MGCP or H.323. However, the open SIP architecture constitutes the provided services vulnerable to various attacks, similar to those currently existing in Internet. The lack of a formal way to describe VoIP vulnerabilities hinders the development of tools that could be utilized for identifying such vulnerabilities or for testing the security level of the offered services, in both cases the tools being independent from a specific implementation. This paper introduces such a formalization for SIP-based VoIP services, utilizing ontologies, facilitating an extensible description of known SIP security vulnerabilities that can be employed in a real environment for testing or intrusion detection purposes.

Keywords: Voice over IP; Intrusion detection system; Attack description; Session initiation protocol; Ontology

1. Introduction

Voice over IP (VoIP) telephony services suffer from various types of attacks and vulnerabilities mainly due to the utilization of an open environment like Internet. Such problems have been already described in [1,2] and they affect either the signaling protocols like SIP, H.323, MGCP, or the transport protocols like RTP. Besides, these technologies have not been designed with security features/functionalities in mind. Thus one of the main challenges for the telecommunication providers is to identify and prevent such security flaws.

A meta-model that could be used for the identification of non-legal behavior and for security testing could result from a formal representation of the protocol, performed on the basis of its operational characteristics and specifically in terms of: (a) the way it processes messages and (b) the behavior of a legitimate user (message flow). However, such formalization does not exist in traditional Intrusion Detection Systems (IDS) nor it is utilized for security testing. More specifically existing IDSs can identify and consequently protect only against potential intrusions that are represented in accordance to a particular classification and signature language. However, the description in these languages is not easily extensible and they cannot be applied to non-homogenous architectures. Moreover, their semantics are often vague and lack of any formal logic. For instance traditional IDSs such as SNORT [3] and BRO [4] can identify several attacks but there is no a common way to describe such security flaws in the systems. Considering that VoIP is a real time service that requires high availability and robustness, the formalization of VoIP protocols would provide a valuable tool for the identification of security flaws in VoIP architectures.

Ontologies could be considered as a model capable of providing the required formalization and the powerful constructs that include machine interpretable definitions of the concepts within a specific domain and the relation between them. Gruber [5] mentions that one of the most common goals for developing ontologies is for sharing the understanding about the structure of information among people or software agents. Consequently the introduction of

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

0140-3664/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2006.12.023

ontologies in VoIP architectures could contribute towards more robust infrastructures, as it can provide a common-shared description for any type of attack, independently from the specifics of the system implementation. To the best of our knowledge there is only limited literature on the utilization of ontologies for attack description or for their use in existing IDSs [6,7], while there is no literature at all addressing the issue of engaging a security ontology in VoIP environments.

Specifically this work focuses on modeling the security flaws of Session Initiation Protocol (SIP) [8], utilizing an ontology formalization that can be applied either as a countermeasure against attacks on SIP based VoIP services or for testing the security robustness of SIP-VoIP infrastructure.

The remainder of this paper is structured as follows. Section 2 introduces some background information concerning SIP messages and security vulnerabilities in SIP, while Section 3 focuses on ontology description of SIP vulnerabilities. Section 4 presents how such ontology can be used for detecting potential attacks or for testing the robustness of the corresponding infrastructure. Finally Section 5 concludes the paper giving some pointers for future work.

2. Background

2.1. General description of SIP messages

Among others, one of the most important SIP advantages is the inheritance of HTTP structure. According to RFC 3261 [8] a SIP message can be either a request or a response (depending on the first line structure), followed by the appropriate headers which provide specific details about the message, like for instance caller (from), callee (to), routing information etc, and finally the message body that describes the request/response. It is noted that the message body is optional and its existence depends on the type of request (REGISTER, INVITE, etc.). Fig. 1 provides an example of a typical SIP-INVITE request.

Every time a user receives such a request (callee) he is responsible for processing it and for generating the appropriate response message. The only difference of the response message, as compared to the request, lies in the *first line*, where it includes a status code that depends on the user behavior. For example in the case where the callee accepts the call the status code generated is 200 OK, as illustrated in the Fig. 2.

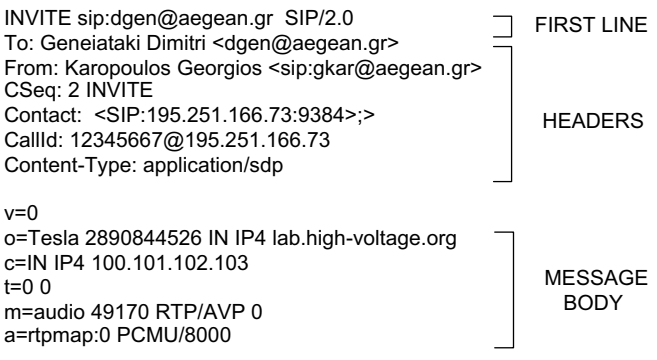

Fig. 1. Typical SIP-INVITE message.

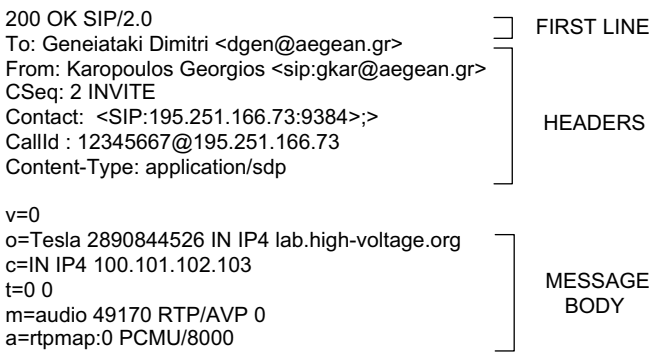

Fig. 2. Typical SIP OK response message.

2.2. Overview of SIP's security flaws

SIP seems to overwhelm all the other signaling protocols that have been designed for Internet telephony, mainly due to the fact that it has been adopted by various standardization organizations (i.e., IETF, ETSI, 3GPP) as the protocol for both wireline and wireless world in the Next Generation Networks (NGN) era. As a result an attacker will focus on exploiting SIP vulnerabilities, especially since besides the advantages originating from the SIP HTTP inheritance, it is clear that SIP inherits HTTP vulnerabilities. For instance a malicious user may generate a malformed message for testing the robustness of a web server and ultimately its conformance to the HTTP protocol. Similar attacks – tests can be launched against SIP servers as well. The PROTOS project [10] has made great strides to identify certain subclasses of malformed input to test a SIP server. This kind of attack can be successfully recognized if SIP servers feature a mechanism that can check the message syntax, based on the SIP grammar. More details about such an identification scheme can be found in [11].

On the other hand, in the case of flooding attack, the attacker may utilize either a well formed or a malformed message trying to cause Denial of Service (DoS). Such an attack can either originate from a single source or simultaneously from several different sources. Consider the possibility of an attacker who generates numerous "call initiation" messages (INVITEs), aiming to harm the availability–reliability of the offered telephony services. To defend against such an attack, at least as far as the single source flooding case is concerned, techniques similar to those applied to Internet applications can be adopted. For example a simple rule would be to specify a threshold for the number of invitations that can be generated from a client at a given period of time. If a client (attacker) exceeds

that threshold, the VoIP-IDS will issue the appropriate alert that will trigger the prevention mechanism to block the identified source.

Both of the aforementioned attacks can be launched either from insiders or from outsiders with the same level of easiness. The main consequence, in case that such an attack is successful, is on the availability and reliability of the offered service which, in the worst case, may end up at a DoS state. Table 1 summarizes such security flaws in SIP.

It should be stressed that in addition to the aforementioned security flaws, there are also attacks, that explore session management and application level vulnerabilities, like signaling attacks [9]. However, such attacks is outside of the scope of the current work, the reason being that in the ontology description the given emphasis is on malformed messages and flooding attacks.

3. Ontology representation of SIP security flaws

Generally speaking any type of attack, irrespective of the application that it aims at, utilizes a protocol trying to cause a specific consequence (e.g., DoS or gain Unauthorized Access) to the corresponding node (target). This general approach has been taken into account throughout the development of the proposed ontology for SIP security flaws. Furthermore, the design of the ontology has been based on the description of the security problems presented in Section 2, focusing to the exploitation of a malformed SIP message. Fig. 3 illustrates the general structure of the proposed ontology. Specifically any SIP attack employs a SIP message that is forwarded to a target node trying to cause a specific consequence. The following subsections describe the "SIP Message" and "SIP Attack" ontology parts correspondingly.

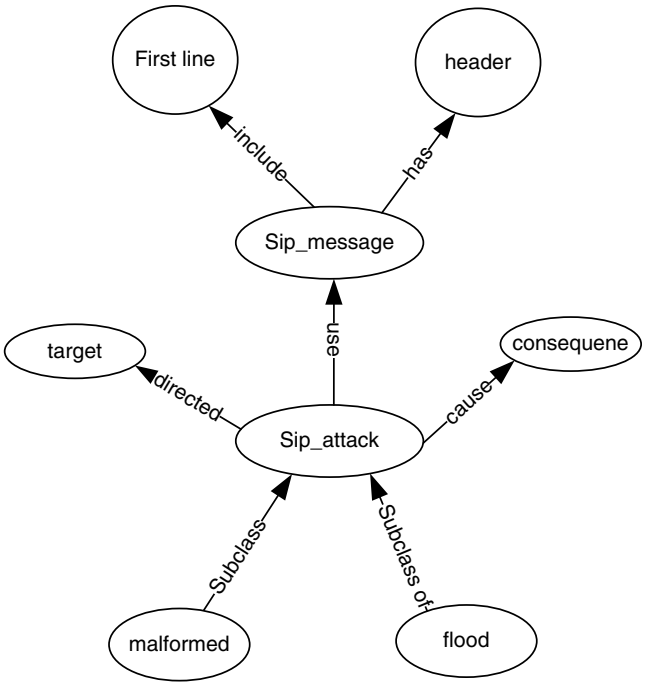

Fig. 3. General description of a "SIP security flaws" ontology.

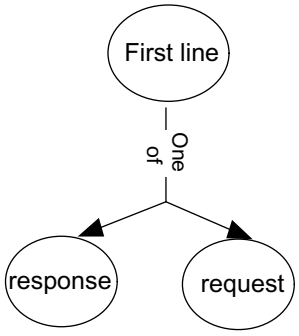

Fig. 4. Ontology first line structure.

3.1. SIP message ontology description

Fig. 3 reveals that one of the main ontology parts is that of the "SIP message", in which it models its structure, in the form of a request or a response (see Fig. 4), including the appropriate headers as described previously in Section 2.1 (see Figs. 1 and 2).

The ontology classes used to represent the SIP message, are the following:

(a) SIP_MESSAGE
(b) FIRST_LINE
(c) HEADER

The above representation follows the SIP message description specified in RFC 3261 [8]. Specifically the FIRST_LINE class distinguishes SIP messages in requests or responses. Each request is described by a method followed by the corresponding Uniform Resource Identifier (URI), which specifies uniquely the requested resource (see Fig. 1). The methods used to describe requests and should therefore exist in the ontology description, are the following: REGISTER, INVITE, SUBSCRIBE BYE, ACK, CANCEL OPTIONS. The URI in the ontology description provides a rule that must be applied to any

Table 1
Attacks in SIP

Attack	Activity (A)ctive/(P)assive	Location (I)nternal/(E)xternal	Source (S)ingle/(M)ulti	Affected security issue	Consequences
Flood	A	I–E	S–M	Availability–reliability	DoS
Malform	A	I–E	S–M	Availability–reliablity	DoS

SIP entity for checking the validity of the incoming request/response message. A rule corresponds to the description of the URI that must follow any SIP message as illustrated in the SIP grammar. Fig. 5 provides, as an example, the description of the REGISTER request (*first_line* tag), which utilizes the REGISTER method and provides the rule for REGISTER URI. The specific rule specifies that the URI must follow the structure of a conventional IP address, providing additional information about the transport protocol and the version of the SIP stack.

As already mentioned in Section 2.1, the SIP responses follow similar structure to that of SIP requests. The main difference lies in the fact that the first line is replaced by the status code and the corresponding description. Fig. 6 demonstrates a SIP response description in the ontology "encoding".

The second part of a SIP message is that with the headers that describe the request/response. As implied by the name, the SIP_HEADER class corresponds to the representation of the SIP headers in the ontology description. It includes two main properties: the header name and the rule that must be checked for deciding whether the header is valid or not. Similarly to the URI rules, the header rule depicts the SIP grammar utilizing a regular expression. For instance, Fig. 7 illustrates an example of a 'TO' header description, including the name of the header and the corresponding rule as described in RFC 3261 [8].

3.2. SIP attack ontology description

The ontology description illustrated in Fig. 3, relates to two types of attacks (malformed messages and single flooding) that can be launched against a SIP node. The general description of such attacks highlights the fact that they use a SIP message that is being forwarded to a specific target trying to cause a specific consequence.

The MALFORMED class corresponds to the attacks employing incoming SIP messages that do not conform to the SIP grammar. Specifically any incoming message that instantiates the SIP message part of the ontology (as described in Section 3.1) but is not found to be consistent with the corresponding description, is categorized as a malformed message. Subsequently, in this formalization a malformed message is the complement of a well-formed message and it does not require any 'special' modeling.

```
<first_line ID="REGISTER">
   <method rdf:resource="#REGISTER"/>
    <uri>
     \s+((((\d{1,3}[.]){3,3}\d{1,3}(:\d{1,5}))
     (;transport[=].+)*\s+(SIP[/]\d[.]\d))|((SIP[/]\d[.]\d)
     \s+(\d{3})\s+.+)\s*
     </uri>
</first_line>
```

Fig. 5. First line description for SIP REGISTER.

```
< first_line ID="RESPONSE">
   <method rdf:resource="#RESPONSE"/>
    <uri>
     \d{1,3}\s+\w+\s+(SIP[/]\d[.]\d)\s+
    </uri>
</first_line>
```

Fig. 6. First line description for SIP response's.

```
<sip_headers rdf:id="to">
    <name>#to</name>
    <rule>
   \s*((["]*(\w+\s*\w*)*["]*)*\s+((<)*(sip:)((\w+@(\w+[.])+
   \w+)| (((\d{1,3}[.]){3,3}\d{1,3}))(>)*))(\s*;tag=.+)*\s*
    </rule>
</sip_header>
```

Fig. 7. Header description for a 'TO' header.

As far as the FLOOD class is concerned, it is associated with the formalization of single flooding attacks. Generally, in any flooding attack an attacker generates a huge number of messages in order to cause DoS. The generated messages can be either well-formed or malformed. One way to represent such an attack could be the utilization of a specific threshold in regard with: (a) the number of messages that a single SIP client is allowed to send, (b) the number of messages that a SIP client can process in a specific period of time and (c) the corresponding memory consumption of the server.

The TARGET class is utilized to describe the potential SIP component target. The main properties employed for that purpose are the IP address and the port of the corresponding service. Finally, the goal of any attacker launching such an attack is to cause a specific CONSEQUENCE, that is either DoS or unauthorized access, to a SIP node.

3.3. Implementing the ontology

To capture the terms and their meaning, a language must describe object classes and relations between objects, in the domain of discourse. For the description of the SIP security flaws ontology the DAML+OIL ontology language [12] has been utilized. DAML+OIL does not only provide a schema language but it also supports the general representation of knowledge. Consequently, a representation of a resource, utilizing such a description, could be interoperable, shareable and understandable among different parties. Clearly, the proposed ontology can be also described/implemented through other ontological languages like OWL [16], based on RDF schema. Even though OWL is a more expressive ontology language, as compared to DAML+OIL, the latter covers the ontological needs for describing security flaws in SIP. Besides, the DAML+OIL description can be easily converted to OWL.

As far as other ontological approaches are concerned, like Rei [17], KAoS [18], we believe that they are not suitable for describing the security flaws of a specific protocol. Rei is a policy framework that integrates support for policy

specification, analysis and reasoning, while KAoS is a collection of services and tools that allow for the specification, management, conflict resolution, and enforcement of policies.

The DAML description of the aforementioned ontology components can be found in Appendix A of this paper.

4. Ontology application

The presented ontology can be applied in a real environment either for testing purposes or for identifying security problems, like the presence of malformed messages and single source flooding attacks. As an example, Fig. 8 depicts a SIP-IDS architecture, based on the proposed ontology. It is stressed that the same ontology description can be utilized for testing the robustness of the provided service. The only difference lies in the fact that for security testing the "testing user" utilizes the complement of the ontology description to generate malicious traffic. For intrusion detection purposes the ontology description is used as is. The rest of this section focuses on an IDS architecture based on the ontology description.

The main components of the proposed IDS architecture are:

(a) The Ontology Server: stores the SIP ontology security descriptions.
(b) The SIP-IDS server: responsible for identifying illegal traffic based on the ontology description.
(c) The SIP based VoIP Server: responsible for the session management.

Whenever the SIP-IDS is initialized it requests an instance description from the ontology server. The SIP-IDS can be updated at any time by asynchronously receiving, from the ontology server, new security flaw descriptions (see Fig. 9). Thus every time that the ontology

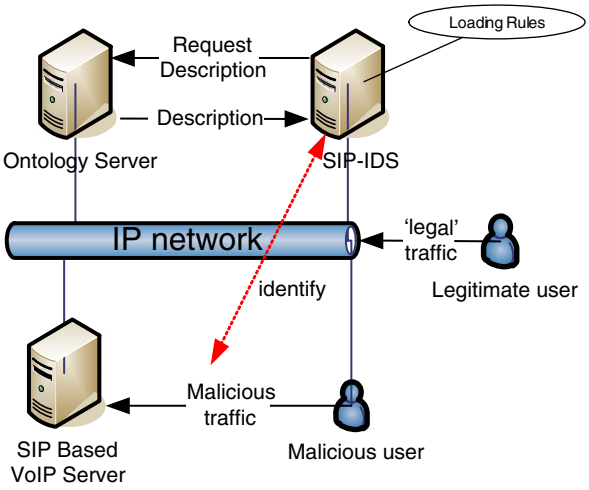

Fig. 8. Ontology based IDS architecture.

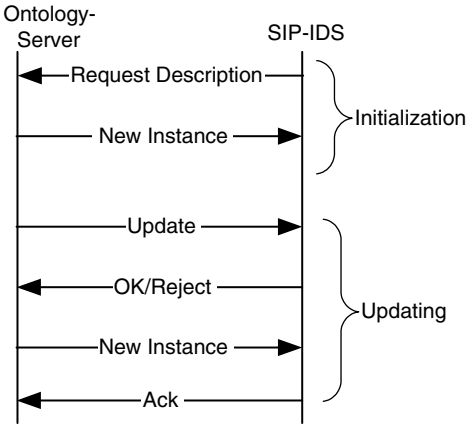

Fig. 9. Asynchronous communication for instance exchange.

description is updated it is also necessary to update the SIP-IDS.

Specifically for the proposed architecture, the Ontology-Server generates an *Update* message in order to inform the SIP-IDS for the existence of an updated instance version. The SIP-IDS is responding with an *OK* or *REJECT* message according to whether it accepts to update the current instance or not. This decision depends on the policy of the specific realm. If the SIP-IDS responds with an *OK*, the Ontology Server sends back the current instance description and as a final step the SIP-IDS acknowledges the receipt of the instance. It is stressed that during initialization or/and updating phase, when the SIP-IDS receives the SIP ontology instance it stores it in a protected file, parses it and "loads" the rules in memory. Fig. 9 depicts the aforementioned scheme. The underlying protocol for realizing this exchange scheme utilizes TCP/IP, even though the same scheme could be also implemented through the FIPA [19] Agent Communication Language. Finally, in order to avoid any 'interference' during the exchange of instances, the underlying communication should be encrypted either with TLS [13] or IPSec [14].

Let us consider a case where an administrator wishes to protect a SIP server against REGISTER malformed message attacks. In order for the SIP-IDS to provide such a protection, it is necessary the ontology server to be presented with an instance of a SIP_MESSAGE class which describes the structure of the REGISTER message. This instance of the SIP_MESSAGE (full listing can be found at Appendix A) follows the ontology description, as presented in Section 3. Furthermore, it must be noted that the instantiation of a SIP message uses the rules that are applicable to the specific message.

Fig. 10 illustrates the instantiation of a SIP REGISTER message. In this example, the *sip_message* tag gives the structure of a REGISTER message, including the REGISTER first_line and the headers that describe this message which are: *CSEQ, FROM, TO*. The headers that will be included in the instantiation of the REGISTER message are depended on the security administrator of the corre-

```
<sip_message id="sip_register">
    <first_line rdf:resource="#REGISTER" id="1"/>
     <header rdf:resource="#CSEQ"/>
     <header rdf:resource="#from"/>
     <header rdf:resource="#to"/>
     <target rdf:resource="#target1"/>
</sip_message>
```

Fig. 10. SIP message class instantiation.

sponding realm. Headers which are not included in the description are simply ignored from the SIP-IDS. The target tag specifies the server that is responsible for the management of this kind of messages in our case the REGISTER messages.

Every resource appearing in the ''sip_message'' instance must exist in the ontology description. Consequently, in the ontology description must be also included: (a) the *first_line* REGISTER, (b) headers *CSEQ, FROM, TO* and (c) the *target1* resources correspondingly. Fig. 11 presents the description of the aforementioned resources.

Consider now a case where a malicious user attempts to register the user ''*dgen*'' by sending to the VoIP server the message shown at Fig. 12. The SIP-IDS parses this

```
<first_line ID="REGISTER">
          <method rdf:resource="#REGISTER"/>
<uri>
\s+((((\d{1,3}[.]){3,3}\d{1,3}(:\d{1,5}))
(;transport[=].+)*\s+(SIP[/]\d[.]\d))|((SIP[/]\d[.]\d)\s+(\d{3})\s+.+)\s*
</uri>
</first_line>
<sip_message_headers rdf:id="to">
          <name>#to</name>
          <rule>\s*((["]*(\w+\s*\w*)*["]*)*\s+
          ((<)*(sip:)((\w+@(\w+[.])+\w+)
          |(((\d{1,3}[.]){3,3}\d{1,3}))(>)*))(\s*;tag=.+)*\s*
          </rule>
</sip_message_headers>
<sip_message_headers rdf:id="from">
          <name>
          #from
          </name>
          <rule>\s*(["]*(\w+\s*\w*)*["]*\s+
          ((<)*(sip:)((\w+@(\w+[.])+\w+)|
          (((\d{1,3}[.]){3,3}\d{1,3}))(>)*))(\s*;tag=\w+)*(.+)*\s*
          </rule>
</sip_message_headers>
<sip_message_headers rdf:id="CSEQ">
          <name>
          #CSEQ
          </name>
          <rule>
          ^\s*(CSeq:)\s*\d+\s+\b(register)\b\s*</rule>
</sip_message_headers>
<target Id="#target1">
          <ip_address>
          195.251.161.143
          </ip_address>
          <port>
          5060
          </port>
</target>
```

Fig. 11. First line and header ontology instantiation.

```
REGISTER null SIP/2.0
Via: SIP/2.0/UDP 195.251.161.142;
CSeq: 4507 REGISTER
To:  <sip:dgen@195.251.161.143>
Expires: 900
From: <sip:dgen@195.251.161.143>
Call-ID: 694128268@195.251.161.142
Content-Length: 0
User-Agent: kphone/4.1.0
Event: registration
Allow-Events: presence
```

Fig. 12. Malformed register message.

message, line by line, and applies the rules found at the first line and the corresponding headers instances (see Fig. 11).

Clearly the validation of the above message fails, as it does not include the appropriate URI in the incoming register (refer to the first line of register in Figs. 10 and 11). In cases where there is no rule for a particular header this specific header is ignored. Furthermore if a REGISTER message is directed to a SIP server that is different form the specified target, then this message will be discarded. Finally, it should be emphasized that a critical issue for such security mechanisms is the delay that they introduce, since such delays may seriously affect real-time communication and services, like VoIP. In [15] it is demonstrated that the overheads introduced by the proposed signature based IDS are negligible.

5. Conclusion and future work

The availability and reliability of a real time processing system, like VoIP, is considered crucial for its successful operation in an open environment like the Internet. The improved security level of the offered services is among the important factors that can be utilized for increasing availability and reliability. The formal representation of the protocol's structure and logical behavior can be a valuable tool for security testing and intrusion detection. In this paper such a formalization, using ontologies, for the SIP protocol is presented. It is the authors' opinion that the same ontology description, with some extensions, could be utilized not only for the SIP protocol but also for other signaling protocols. For instance, an extended ontology description that includes all known security flaws in TCP/IP stack (e.g., spoofing, TCP-SYN or flooding attacks, etc.) would enable the construction of systems capable of defending against most known attacks.

Another important aspect of such security mechanisms is their ability to communicate among heterogeneous networks. Currently we are investigating techniques for exchanging ontologies among heterogeneous VoIP systems. The objective is to improve the security level of services provided by cooperation of more than one party, but also to support VoIP providers to cooperate for defending against known and unknown threats.

Appendix A

```
<daml:Class rdf:ID=SIP_MESSAGE>
   <daml:subclassof>
      <daml:Restriction>
         <daml:onProperty rdf:resource=“first_line”/>
         <daml:hasClass rdf:resource=“sip_ first_line”/
         >
      </daml:Restriction>
   </dam:subclassof>
   <daml:subclassof>
      <daml:Restriction>
         <daml:onProperty rdf:resource=“first_line”/>
         <daml:cardinality>1</daml:cardinality>
      </daml:Restriction>
   </daml:subclassof>
   <daml:subclassof>
      <daml:Restriction>
         <daml:onProperty rdf:resource=“headers”>
         <daml:mincardinality>3</daml:cardinality>
      </daml:Restriction>
   </daml:subclassof>
   <daml:subclassof>
      <daml:Restriction>
         <daml:onProperty rdf:resource=“headers”>
         <daml:range rdf:resource=“sip_headers”>
      </daml:Restriction>
   </daml:subclassof>
</daml>

<daml:Class rdf:ID=sip_first_line>
   <daml:disjointUnionOf
   parseType=“daml:collection”>
      <daml:Class rdf:about=“request”/>
      <daml:Class rdf:about=“responses”/>
   </daml:disjointUnionOf>
</daml>

<daml:DatatypeProperty rdf:ID=“uri”>
   <daml:domain rdf:resource=“#sip_first_line”/>
   <rdf:range rdf:Resource=“string”/>
</daml:DatatypeProperty>

<daml:Class rdf:ID=“request”>
</daml:Class>

<daml:objectProperty ref:ID=“used_method”>
   <daml:domain resource=“#request”/>
   <daml:range rdf:resource=“#methods”/>
</daml:objectProperty>

<daml:Class rdf:ID=“methods”>
   <daml:oneOf ref:parseType=“Collection”>
      <daml:Thing rdf:about=“REGISTER”>
      <daml:Thing rdf:about=“INVITE”>
      <daml:Thing rdf:about=“SUBSCRIBE”>
      <daml:Thing rdf:about=“BYE”>
      <daml:Thing rdf:about=“ACK”>
      <daml:Thing rdf:about=“CANCEL”>
      <daml:Thing rdf:about=“OPTIONS”>
   </daml:oneof>
</daml>

<daml:Class rdf:ID=sip_headers>
</daml>
<daml:DatatypeProperty rdf:ID=“header_name”>
   <daml:domain rdf:resource=“#sip_headers”/>
   <rdf:range rdf:Resource=“string”/>
</daml>
<daml:DatatypeProperty rdf:ID=“rule”>
   <daml:domain rdf:resource=“#sip_ headers”/>
   <rdf:range rdf:Resource=“string”/>
</daml>

<daml:Class rdf:ID=attack>
</daml:Class>
<daml:ObjectProperty rdf:ID=“attack_utilize”>
   <daml:domain rdf:resource=“#attack”/>
   <rdf:range rdf:Resource=“#SIP_MESSAGE”/>
</daml:ObjectProperty>
<daml:ObjectProperty rdf:ID=“attack_target”>
   <daml:domain rdf:resource=“#attack”/>
   <rdf:range rdf:Resource=“#target”/>
</daml:ObjectProperty>

<daml:Class rdf:ID=“malformed”>
   <rdfs:subclassof resource=“#attack”/>
   <rdfs:subclassof>
      <daml:complementof>
         <daml:Class rdf:resource=#sip_message/>
      </daml:complementof>
   </rdfs:subclassof>
</daml:Class>

<daml:Class rdf:ID=“flood”>
   <rdfs: subclassof resource=“#attack”>
</daml:Class>
<daml:DatatypeProperty rdf:ID=“threshold”>
   <daml:domain rdf:resource=“#singleflood”/>
   <rdf:range rdf:Resource=“number”/>
</daml:DatatypeProperty>
<daml:DatatypeProperty
rdf:ID=“memoryconsumption”>
   <daml:domain rdf:resource=“#singleflood”/>
   <rdf:range rdf:Resource=“number”/>
</daml:DatatypeProperty>
<daml:DatatypeProperty rdf:ID=“duration”>
   <daml:domain rdf:resource=“#singleflood”/>
   <rdf:range rdf:Resource=“number”/>
</daml:DatatypeProperty>
```

References

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

[2] D. Geneiatakis, A. Dagiouklas, G. Kambourakis, C. Lambrinoudakis, S. Gritzalis, S. Ehlert, D. Sisalem, Survey of Security Vulnerabilities in Session Initiation Protocol, IEEE Communications Surveys and Tutorials, vol. 8 (3), IEEE Press, 2006, pp. 68–81.

[3] SNORT the defacto standard for intrusion detection/prevention. Available from: <www.snort.org>.

[4] BRO Intrusion Detection System. Available from: www.bro-ids.org.

[5] T. Gruber, Towards principles for the design of ontologies used for knowledge sharing, Formal Ontology in Conceptual Analysis and Knowledge Representation, Kluwer, 1993.

[6] J. Undercoffer, A. Joshi, A. Pinkston, Modeling computer attacks: an ontology for intrusion detection, Lecture Notes in Computer Science 2820 (2003) 113–135.

[7] J. Undercoffer, A. Joshi, T. Finin, J. Pinkston, A target-centric ontology for intrusion detection, in: 18th International Joint Conference on Artificial Intelligence, Acapulco, Mexico. Available from: <http://citeseer.ist.psu.edu/568003.html>.

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

[9] T. Dagiuklas, D. Geneiatakis, G. Kambourakis, D. Sisalem, S. Ehlert, J. Fiedler, J. Markl, M. Rokos, O. Botron, J. Rodriguez, J. Liu, General Reliability and Security Framework for VoIP Infrastructures. Available from: <http://www.snocer.org>, Aug. 2005.

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

[11] D. Geneiatakis, G. Kambourakis, T. Dagiuklas, C. Lambrinoudakis, S. Gritzalis, A framework for detecting malformed messages in SIP networks, in: Proceedings of 14th IEEE Workshop on Local and Metropolitan Area Networks (LANMAN), September 2005, Chania, Crete, Greece, 2005.

[12] D.L. McGuinness, R. Fikes, J. Hendler, L.A. Stein, DAML+OIL: An Ontology Language for the Semantic Web, In IEEE Intelligent Systems 17 (5) (2002) 72–80.

[13] T. Dierks, C. Allen, The TLS Protocol Version 1.0, RFC 2246, January 1999.

[14] S. Kent, R. Atkinson, Security architecture for the Internet Protocol, RFC 2401, November 1998.

[15] D. Geneiatakis, G. Kambourakis, C. Lambrinoudakis, A. Dagiouklas, S. Gritzalis, A framework for protecting a SIP-based infrastructure against malformed message attacks, Comput. Netw. (2007), doi:10.1016/j.comnet.2006.11.014.

[16] OWL WEb Ontology Language Guide. Available from: <http://www.w3.org/TR/owl-guide/>.

[17] L. Kagal, Rei: A Policy Language for the Me-Centric Project, HP Labs Technical Report, HPL-2002-270, 2002.

[18] M. Johnson , P. Chang, R. Jeffers, J. Bradshaw, M. Breedy, L. Bunch, S. Kulkarni, J. Lott, N. Suri, A. Uszok, V.W. Soo, KAoS Semantic Policy and Domain Services: an application of DAML to web services-based grid architectures, in: Proceedings of the AAMAS 03 workshop on Web Services and Agent-Based Engineering, Melbourne, Australia, July 2003.

[19] Foundation for Intelligent Physical Agent, Specification for Agent Communication. Available from: <http://www.fipa.org>.

Dimitris Geneiatakis was born in Athens, Greece, in 1981. He received the five-year Diploma in information and communication systems in 2003, and the M.Sc. in security of information and communication systems in 2005, both from the department of Information and Communications Systems Engineering of the University of Aegean, Greece.. His current research interests are in the areas of Security mechanisms in Internet telephony, Smart Cards and Network Security. He is an author of several refereed papers in international scientific journals and conference proceedings. Mr. Dimitris Geneiatakis is a member of the Technical Chamber of Greece.

Costas Lambrinoudakis 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 form the University of London (UK). Currently he is an Assistant Professor at the Department of Information and Communication Systems of the University of the Aegean. His current research interests include: Information Systems Security, Smart Cards and Computer Architectures. He is an author of several refereed papers in international scientific journals and conference proceedings. He has participated in many national and EU funded R&D Projects. He has served on program and organizing committees of national and international conferences on Informatics and he is a reviewer for several scientific journals.