Realtime DDoS detection in SIP Ecosystems: Machine Learning tools of the trade

Zisis Tsiatsikas[1], Dimitris Geneiatakis[2], Georgios Kambourakis[1], and Stefanos Gritzalis[1]

[1] Dept. of Inform. and Comm. Systems Engineering, University of the Aegean, Karlovassi, Greece
[2] Electrical and Computer Engineering Department, Aristotle University of Thessaloniki, GR541 24 Thessaloniki, Greece

Abstract. Over the last decade, VoIP services and more especially the SIP-based ones, have gained much attention due to the low-cost and simple models they offer. Nevertheless, their inherently insecure design make them prone to a plethora of attacks. This work concentrates on the detection of resource consumption attacks targeting SIP ecosystems. While this topic has been addressed in the literature to a great extent, only a handful of works examine the potential of Machine Learning (ML) techniques to detect DoS and even fewer do so in realtime. Spurred by this fact, the work at hand assesses the potential of 5 different ML-driven methods in nipping SIP-powered DDoS attacks in the bud. Our experiments involving 17 realistically simulated (D)DoS scenarios of varied attack volume in terms of calls/sec and user population, suggest that some of the classifiers show promising detection accuracy even in low-rate DDoS incidents. We also show that the performance of ML-based detection in terms of classification time overhead does not exceed 3.5 ms in average with a mean standard deviation of 7.7 ms.

Keywords: VoIP; SIP; DoS; DDoS; Machine Learning; Evaluation.

1 Introduction

Throughout the last decade, Voice over IP (VoIP) services are gaining increasing attention due to the multiple advantages they offer in comparison to those provided by Public Switched Telephone Network (PSTN). Based on current market reports, VoIP is blooming and its market share is estimated to reach 130$ billion until 2020 [1]. On the other hand, as VoIP services rely on the open Internet, providers need to ensure availability levels similar to PSTN. This means that among other well-documented threats [2–5] they need to cope with resource consumtion attacks namely Denial os Service (DoS) as well as their distributed form (DDoS) that cause service disruptions and sometimes even complete outages. This is of high importance especially for critical voice services *e.g.*, emergency numbers. This threat is further aggravated as the current predominant VoIP signaling protocol, namely Session Initiation Protocol (SIP), can be easily exploited by an attacker. This is mainly due to SIP text nature that allows the aggressor to straightforwardly craft and send large volumes of SIP requests toward its victim with the aim to paralyze it. The perpetrator is also able to exercise more clever attacks, including low and slow ones [6] in order to consume a considerable amount of VoIP server's resources and network bandwidth, and thus degrade the quality of the service.

To cope with this threat, several SIP-oriented intrusion detection and prevention systems have been presented in the literature so far. Focusing on proposals to defend against DoS in these environments, one can identify simple statistical schemes as those given in [7, 8]. In this category of solutions detection relies on different network statistics, including incoming traffic rate, and uses a predefined threshold above which the received traffic is classified as malicious. It is obvious though that such a solution cannot protect effectively SIP services against low-rate DoS attacks, as it is cumbersome to constantly adapt itself, say, by recalculating the underlying threshold to reflect the ongoing characteristics of the attack.

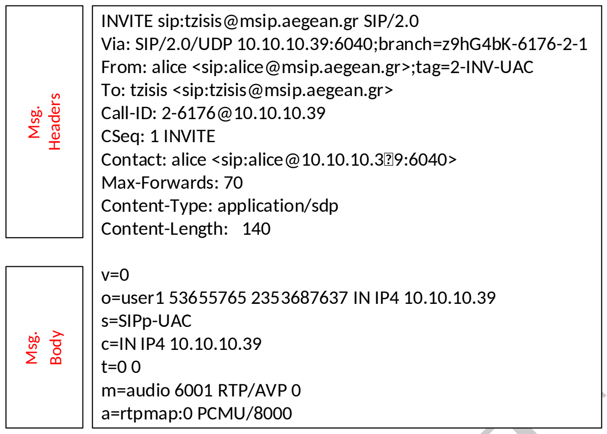
Further, most of the existing solutions do not consider SIP different inherent features and characteristics which can be exploited by an attacker to launch DoS, while the majority of them are privacy-invasive. A comprehensive analysis of existing protection solutions against DoS in SIP can be found in [9]. Therefore, as the aggressors become more sophisticated, there is a need for advanced (D)DoS detection methods that are able to automatically adjust their behavior to the attack traffic patterns. A way to achieve this is to use well-established classifiers from the Machine Learning (ML) toolbox.

Our contribution: The paper at hand assesses the potential of using techniques borrowed from the ML realm to detect (D)DoS incidents in SIP services. In contrast to our previous work [10], the evaluation of the various classifiers is done in realtime in the SIP proxy. Our experiments involve 5 different well-known classifiers and a large variety of attack scenarios ranging from simple DoS to slow and high-rate DDoS. The evaluation is done in terms of both detection accuracy and proceesing time. The results show that the introduced overhead in the SIP server is negligible while the average detection accuracy for the best-performing algorithms fluctuates between ≈89% and 92% depending on the scenario.

The remainder of the paper is organized as follows. The next section provides an overview of SIP architecture and briefly analyses the threat model. Section 3 describes the methodology and steps followed throughout our experiments. Section 4 presents the results. Section 5 overviews the related work. The last section concludes and provides directions to future work.

2 SIP Architecture & Threat model

SIP is a text-based protocol with syntax similar to that of HTTP. As shown in fig. 1, a SIP message (request in this case) consists of two basic parts. The upper one corresponds to the message headers and carries information in regards to the sender (caller) and the recipient (callee) of the message. The second part


Msg. Headers
Msg. Body

```
INVITE sip:tzisis@msip.aegean.gr SIP/2.0
Via: SIP/2.0/UDP 10.10.10.39:6040;branch=z9hG4bK-6176-2-1
From: alice <sip:alice@msip.aegean.gr>;tag=2-INV-UAC
To: tzisis <sip:tzisis@msip.aegean.gr>
Call-ID: 2-6176@10.10.10.39
CSeq: 1 INVITE
Contact: alice <sip:alice@10.10.10.3□9:6040>
Max-Forwards: 70
Content-Type: application/sdp
Content-Length:   140

v=0
o=user1 53655765 2353687637 IN IP4 10.10.10.39
s=SIPp-UAC
c=IN IP4 10.10.10.39
t=0 0
m=audio 6001 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

Fig. 1. A typical SIP Invite request

is known as the message body and carries the media details. Communications resources in SIP are assigned a SIP Uniform Resource Identifier (URI), e.g., with reference to the first line of fig. 1, sip:tzisis@msip.aegean.gr. Every SIP message is proccessed by the appropriate SIP component. A basic SIP infrastructure consists of:

– **SIP User Agent (UA)** - Represents the end points of the SIP protocol, that is, the User Agent Client (UAC) and the User Agent Server (UAS) which are able to initiate or terminate a session using a SIP software or hardware client.
– **SIP Proxy Server** - An intermediate entity which plays the role of the client and the server at the same time. Its task is to route all the packets being sent and received by the users participating in a SIP session. Note that one or more SIP proxies may exist between any two UAs.
– **Registrar** - Handles the authentication and register requests initiated by the UAs. For this reason, this entity stores the user's credentials and UA location information.

For a detailed analysis of SIP architecture the interested reader can consult the corresponding RFC [11].

So far, a plethora of attacks against SIP-based services have been identified in the literature. These include eavesdropping, flooding, SQL injection, manipulation of SIP messages, and so on [2–5, 12]. In this work, we concentrate on resource consumption attacks caused by malicious entities who send a surge of SIP messages against their target, that is, a SIP server or UA. From an attacker's viewpoint, this category of attacks are considered quite straightforward mostly because of the text-based nature of the protocol, the lack of built-in countermeasures, and the existence of open source publicly available attack tools [13]. On the other hand, the impact of such an attack on the target is considerable and may vary from loss of service to entire network paralysis. From a VoIP provider viewpoint, this may result in many dissatisfied customers and loss of profit.

Having in mind all the above, in the following we formulate an attacker-centric threat model. Specifically, we assume that the perpetrator is able to fabricate a SIP message by simply spoofing its headers. The most appropriate SIP requests to achieve such a goal are Invite, Register, and Options [11]. The attacker spoofs certain headers of the message (Via, From, To, Call-ID, Contact) to create a flooding effect towards the victim and obfuscate the forensic signal of the attack. For instance, if the attacker knows the URI of a certain user, is then able to mount a high-rate DoS attack with Invite requests to choke the user's softphone. An indicative example of a device that is prone to such a vulnerability is Cisco SIP Phone 3905 [14]. An alternative attack strategy aims at paralysing critical components of SIP infrastructure, including SIP Proxy or Registrar.

In this paper we only consider the manipulation of Invite messages. However, as already pointed out, the same effect can be achieved with other types of SIP requests.

3 Detection Engine

This section elaborates on the architecture of the proposed IDS, which as observed from fig. 2, is composed of 3 modules. The first module is occupied with the extraction of the required classification features from the headers of the incoming SIP messages. The selected features are forwarded to an anonymisation module, and finally are fed to the classification engine. After a training phase, the latter module can be configured to operate in realtime using an ML algorithm of choice. The subsequent sections elaborate on each of the aforementioned IDS modules.

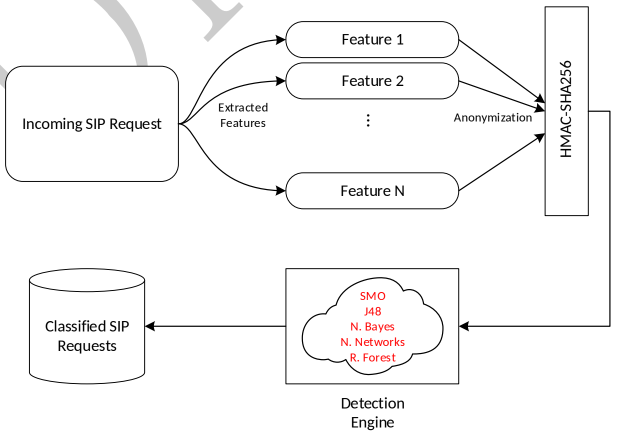


Fig. 2. Detection Engine Architecture

3.1 Feature Exctraction & Anonymization of data

As already mentioned, the feature extraction module operating on the SIP server examines the incoming SIP traffic for any request, say, Invite, Register. Next, the request is parsed to isolate the headers of interest, namely,<Via>, <From>, <To>, <Contact> and <Call-ID>. The extracted headers are anomymised with the help of HMAC-SHA256. The latter aims at preserving end-user privacy in cases where the detection task is outsourced to a third party. Also, it has the dual benefit of preserving the entropy of the original headers and making deanonymization as hard as reversing the HMAC-SHA256. The anonymized features along with its frequency of appearance are stored in a hash table data structure.

As described in Algorithm 1, every time a new (different) feature is extracted from an incoming SIP request a new record is inserted in the hash table with its corresponding value equals to 1. In case an already existing key is insterted in the table the corresponding frequency value is increased by 1. This procedure continues until the number of messages reaches a certain predefined Message Window M_w. For the needs of our experiments we picked arbitrarily a M_w equal to 1,000. Therefore, the detection engine starts the classification process from message 1001. That is, the hash values of the headers of this message will update the corresponding cells of the hash table and the resulting numbers will be fed to the classifier. This process is done in a message-by-message manner for every SIP request arriving after the 1000th.

Obviously, the value of M_w parameter is sure to affect the detection accuracy. Naturally, this parameter can be adjusted by the service provider itself, say, according to the average call rates. Nevertheless, to our knowledge, there is no foolproof approach to formally define this parameter, mainly because it is eminently contextual. That is, it is closely connected to the local characteristics of the service and underlying network. As a result, similar to other anomaly-based approaches, one can follow an error-trial approach to equilibrate between the M_w parameter and the false alarm rate.

3.2 Training and operation

Similar to any other ML-powered approach, the detection engine requires the training of the classifier in order to be able to detect anomalies in the incoming traffic. This means that during service initialization, the classifier of choice must be trained based on some pregenerated training data. The creation of such a training set is up to the service provider because it mainly depends on the particular services it offers and the characteristics of the underlying network. In our case, the training file is produced using the "offline software module" given in [15]. More specifically, for generating the appropriate training file used by the SIP proxy at its initialization phase, we first execute a basic scenario and capture the SIP traffic in a log file. After that, this file is analysed with the previously mentioned module with the aim to generate a new training file ready to be fed to the SIP Proxy. Also, a renewal and/or amendment of the training set is required

Algorithm 1: Extraction of classification features

```
Input: Incoming SIP messages
Output: Classification result
SIPHeaders[N] ← ExtractSipHeader(SIP Request);
for (i=1; i ≤ N; i++) do
    HashedHeader[i] ← HMAC(SIPHeaders[i]);
    if (InsertToHashTable(HashedHeaders[i]) ≠ NULL) then
        GetValueofHashTable(HashedHeader[i])++;
    else
        InsertToHashTable(HashedHeaders[i]);
        SetValueInHashTable(HashedHeader[i]) ← 1;
    end
end
classificationResult ← classify(#HashedHeaders[1],... ,#HashedHeaders[N]);
if ($Mw = 1,000$) then
    TotalMessages ← TotalMessages + $M_w$;
    Re-Initialize(HashTable);
end
```

as soon the network and/or service operating conditions change. In this paper, we opt not to address the two aforementioned issues which are left for future work.

Nevertheless, it should be stressed that the training data must contain both legitimate and attack traffic. This is necessary because in order for the classifier to build a realistic traffic model, the training data must contain classes of both attack and normal traffic. As soon as the training phase is completed, the detection engine starts on the SIP server as a realtime service.

3.3 Implementation

The realtime detection service has been implemented as a plug-in module of the well-known SIP proxy Kamailio [16]. Specifically, the module [15] is written in C programming language and is capable of processing any incoming SIP request as described in the previous subsection.

The feature extraction module stores temporarily the processed data to a hash table, while classification relies on Weka [17] a well-known framework for ML analysis. Given that Weka provides a Java interface, we use Java Native Interface (JNI) [18] to make possible the integration between the feature extraction and classification module. In this way, one can easily configure the employment of the appropriate classifier depending on the requirements at hand.

The implementation has been tested for possible memory leaks following an error-trial approach and monitoring the Linux OS memory consumption under various scenarios. This is because JNI implementation should be protected against potential out-of-memory conditions [19]. As pointed out in section 1, our implementation is freely available [15] for further development and experimentation.

4 Results

This section details on the testbed used and presents the results in terms of both false alarms and processing overhead on the SIP server.

4.1 Test-bed Setup

As illustrated in fig. 3, we employed a virtualised testbed running over an i7 processor at 2.2 GHz. Three Virtual Machines (VMs) were created, each one equiped with 6 GB of RAM. These VMs respectively host the SIP proxy, the UAs, and the attack traffic generator. We created distinct patterns of legitimate, single source DoS and DDoS traffic using *sipp v.3.2* [20], *sipsak* [21], and SIPp-DD [22] tools respectively.

As observed from table 1, twenty two disparate scenarios were created in total to replicate different (D)DoS incidents. Seven basic scenarios were used for training, while the others represent an attack incident. For all the scenarios, an exponential inter-arrival time distribution ($\lambda = 100$) is followed to produce the legitimate traffic. Note that this kind of distribution inherently presents the "lack-of-memory" property. Specifically, this property considers that the probability of a future event (call arrival) is the same regardless of the previous events that took place in a series of time frames. In our case this is analogous to the traffic used for assessing VoIP systems' performance [23].

Furthermore, a range of different call rates has been used in the cases of DoS and DDoS, with the aim to simulate various call rates which may approximate the traffic patterns of a real VoIP provider. For example, as observed in Table 1, the call rate for SN6.1 is given as 20-120, where the first number indicates the call rate of the attack, and the second designates the call rate of the legitimate traffic both occurring in parallel. Keep in mind that for DDoS scenarios about half of the registered users were generating the normal traffic, while the other half were acting maliciously.

4.2 Detection accuracy

In the context of our experiments, we employed 5 well-known classifiers, namely SMO, Naive Bayes, Neural Networks, Decision Trees (J48), and Random Forest. This particular choice has been done because these classifiers present a better detection accuracy when it comes to numerical data [24]. The detection accuracy of each classifier in terms of False Positives (FP) and False Negatives (FN) is estimated and the results are included in Table 2. Also, to ease the reading of the table, the mean, minimum, maximum FN and FP values per classifier and collectively per attack type (DoS/DDoS) are depicted in fig. 4. From this figure it can be seen that FN metric fluctuates between 0.9% and 23.7% for DoS scenarios having an average of 14%, while the corresponding values for DDoS are 2%, 62%, and 16%.

Focusing on Table 2 and on its lines containing the average values of FN metric, one can conclude that SMO produces the worst results and therefore

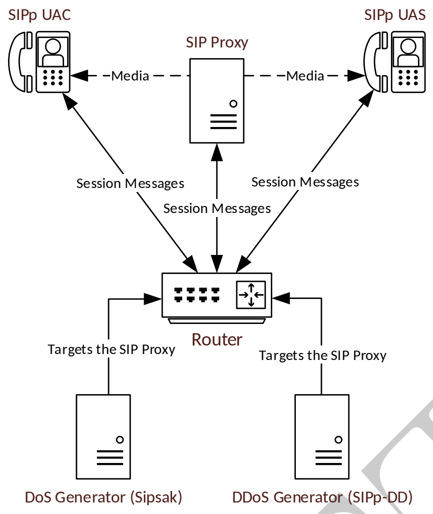


Fig. 3. Deployed Testbed for (D)DoS simulations

should be avoided. The same observation applies to Random Forest but only for DoS scenarios. J48 on the other hand scores an average FN of 11.8% but largely fails in terms of FP (interestingly, the FP metric is almost non-existent for all the algorithms but J48). It can be safely argued that the most reliable classifier across all scenarios and for both FN and FP metrics seems to be Naive Bayes followed by Random Forest. In any case, the FN percentages scored by both the aforementioned algorithms especially for DoS scenarios are not favorable for any real-world IDS. Putting aside SMO and Neural Networks, the rest of the classifiers produce an average FN lesser than 8.5% for DDoS scenarios only. It can be therefore estimated that ML-driven detection shows greater potential in detecting more sophisticated attacks of this kind.

Having a complete view of the results, we consider that further experimentation is needed to obtain a better approximation of the power of ML-driven (D)DoS detection in SIP realms. In this direction, a future work could concentrate on testing more classifiers and tuning the Mw parameter based on the specific needs of the VoIP service provider.

4.3 Performance

Putting aside its effectiveness, the other decisive factor for any IDS is that of performance in terms of service time. This section elaborates on the per message (SIP request) processing time introduced by the realtime detection engine on the SIP server. It is to be noted that all the time measurements included in this section correspond to a worst case scenario as the SIP proxy for all the tests was configured with one serving thread.

Table 1. Description of scenarios

Scen.	Num.of Users	Calls/Sec.	Train Scen.	Type of Attack
SN1	30	2	Yes	-
SN1.1	30	50	-	DoS
SN1.2	30	175	-	DoS
SN1.3	30	350	-	DoS
SN2	30	5	Yes	-
SN2.1	30	20	-	DoS
SN2.2	30	40	-	DoS
SN2.3	30	80	-	DoS
SN3	30	20	Yes	-
SN3.1	30	266	-	DoS
SN4	30	120	Yes	-
SN4.1	30	800	-	DoS
SN5	50	120	Yes	-
SN5.1	50	400	-	DoS
SN5.2	50	1200	-	DoS
SN6	60	20	Yes	DDoS
SN6.1	60	20-120	-	low-rate DDoS
SN6.2	60	120-20	-	high-rate DDoS
SN7	500	100	Yes	DDoS
SN7.1	500	10-200	-	low-rate DDoS
SN7.2	500	100-40	-	high-rate DDoS
SN7.3	500	30-50	-	low-rate DDoS

Fig. 5 illustrates a random snapshot of the processing time introduced by our architecture, while Table 3 includes the average, max, min, and standard deviation processing times per classifier for all the 15 attack scenarios. As observed from the Table, the average processing time remain under 4ms, while it is easily seen that all classifiers follow a similar tendency. As the SIP proxy is configured in single-thread mode it can be safely argued that the induced overhead is negligible. The maximum values of, say, 300 ms contained in both fig. 5 and Table 3 are unique or very scarce and are due to the activation of single-thread mode at SIP proxy side and the traffic pattern of the examined scenario. For instance, while in SN1.2 the normal traffic rate is 2 calls/sec, in SN7.1 is 200 calls/sec resulting to an increment in the classifiers, average processing time.

5 Related Work

Until now, a handful of works discuss the suitability of methods borrowed from the ML community to detect DoS attacks in VoIP realms and particularly in SIP. In the following, we elaborate on each of them starting from those that are designed to work offline.

The authors in [10] provide an offline assessment of 5 different ML classifiers in detecting DoS and DDoS attacks in SIP. This was done by examining the recorded SIP audit trails in a forensic-like manner. Their results obtained under a variety of attack scenarios indicate that ML-powered methods achieve better

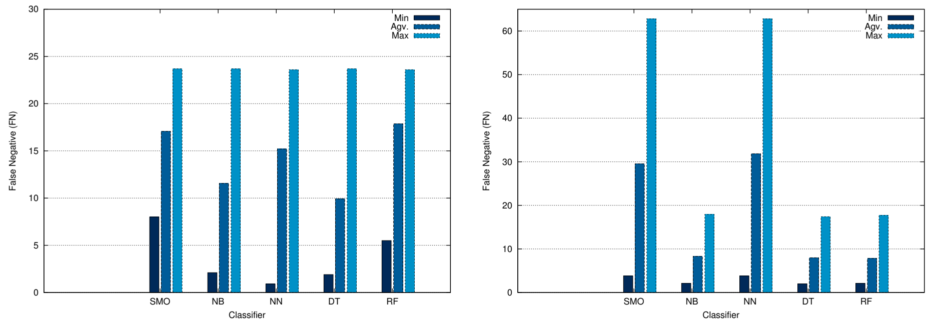


Fig. 4. Dos (left) and DDoS (right) minimum, maximum and average FN percentage values per classifer.

Table 2. Summary of results for all the scenarios (The best performer per scenario in terms of FN is in bold).

SN	**Traffic (Calls)**		**SMO**		**Naive Bayes**		**Neural Networks**		**Decision Trees (J48)**		**Random Forest**	
			FP	FN	FP	FN	FP	FN	FP	FN	FP	FN
	Total Rec.	**Attack Rec.**	%	%	%	%	%	%	%	%	%	%
SN1.1	2.4k	1.2k	0	23.7	0	23.7	0	**23.6**	35.1	23.7	0	**23.6**
SN1.2	2.5k	1.3k	0	22.9	0	**22.8**	0	22.9	33.7	**22.8**	0	23
SN1.3	2.7k	1.5k	0	**21.8**	0	21.8	0	**21.8**	32	22	0	**21.8**
SN2.1	2.7k	1.2k	0	21	0	7.3	0	14	36.9	**6.3**	36.3	19.6
SN2.2	2.7k	1.2k	0	16.1	0	7.3	0	28	29.9	**7.1**	0	9.9
SN2.3	3k	1.3k	0	12.4	0	**5.5**	0	24.8	44.1	14.7	0	27.3
SN3.1	5k	2.3k	0	19.8	0	**6.9**	0	9.1	47.1	7.2	0	20
SN4.1	8k	1.5k	0	8	0	8.9	0	4.7	75.1	**1.9**	0	15.7
SN5.1	7k	1k	0	11.7	0	9.2	0	**0.9**	0	11.2	0	5.5
SN5.2	9k	2.6k	0	12.3	0	2.1	0	2.2	59.9	**2**	0	12.3
Avg.	—	—	0	16.97	0	11.55	0	15.2	39.38	11.8	3.63	17.8
SN6.1	12k	6.6k	0	3.8	0	3.8	0	3.8	0	**3.4**	0	3.7
SN6.2	4.5k	2k	0	18.2	0	17.9	13.7	28.5	0	**17.4**	0	17.7
SN7.1	10.5k	3.3k	0	31.1	0	2.1	0	31.2	0	**2**	0	2.1
SN7.2	7k	2.2k	0	32	0	10.7	0	32.8	0	**10.5**	0	10.6
SN7.3	9k	6k	0	62.6	0	7	0	62.8	0	6.6	0	**5.1**
Avg.	—	—	0	29.5	0	8.3	2.74	31.8	0	7.98	0	7.84

results when compared to those scored by legacy statistical schemes [25, 26]. They also highlight on the fact that some classifiers do achieve satisfactory results even in the case of low-rate flooding attacks. For instance, according to the authors, the Neural Networks classifier succeeds a FP rate of 5.2% and zero FN. Another offline proposal for detecting anomalies in SIP messages is given in [27]. Particularly, among others, the authors employ the Decision Tree (J48) classifier with the aim to expose Invite and Register flooding incidents. To do so, they rely on classification features taken from three SIP message headers, namely <Method>, <To> and <From>. For DoS attacks, they report a detection accuracy of approximately equal to 99.7%.

The work in [28] also reports on the effectiveness of ML methods to cope with flooding incidents in SIP ecosystems. In this case, the authors focus on the first line of 13 different SIP requests and 6 different responses. The authors examine

Table 3. Summary of classification time overhead for all the scenarios (ms)

Classifier	Min.	Max.	Avg.	St. Dev.
SMO	0.08	123.78	3.37	7.73
Naive Bayes	0.15	171.13	3.48	7.76
Neural Networks	0.10	129.20	3.42	7.74
Decision Trees (J48)	0.08	388.01	3.28	7.91
Random Forest	0.08	91.74	3.56	7.72

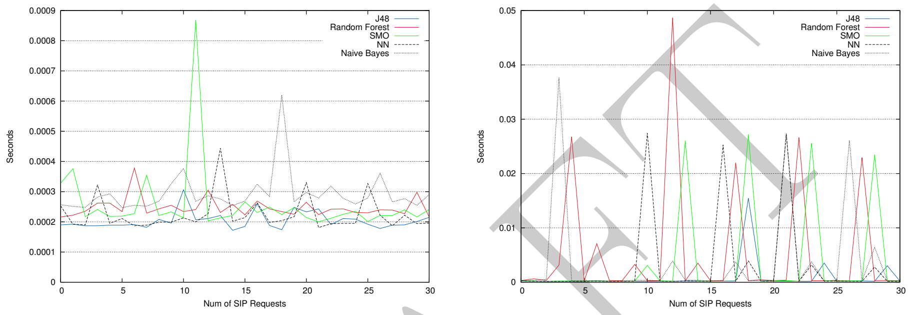


Fig. 5. A random snapshot of the time overhead for scenario SN-1-2 (left) and SN-7-1 (right)

several classifiers, including Evolving Radial Basis Function, Fuzzy AdaBoost Genetic Classifier System, and others. They report sufficing results ranging from 0% to 1% for FP metric and 0% to 97.7% for FN. On the downside, the authors do not consider DDoS, and their experiments were conducted on an artificially created dataset where the simulated attack messages were injected to the normal traffic.

Three real-time detection schemes against Spam over Internet Telephony (SPIT) and flooding attacks are given in [29–31]. The first one deals with the detection of both SPIT and typical flooding attacks carried out via SIP messages. The detection mechanism proposed by the authors capitalizes on 38 distinct classification features and makes use of SVM classifier. The authors report a detection accuracy spanning from 0% to 98.2% depending on the flooding rate. The work in [30] proposes a mechanism that assembles vectors of certain features contained in SIP messages and subsequently feeds it to Naive Bayes and J48 classifiers. More specifically, the spatial features used in this work pertain to the IP address contained in the <From> header and the call ratio of Invite requests. As in their previous work [28], the authors synthetically inject attack messages to the normal SIP traffic in order to estimate the detection accuracy of their proposal. They mention false alarm rates ranging from 0% to 0.7% and 0% to 25% for FP and FN metrics respectively. The third work [31] offers an online intrusion system to combat DoS attacks. The authors argue that their system presents a 99% accuracy, presenting a detection time overhead equal to 152μ. Nevertheless, they neither consider DDoS attacks nor different attack rates in their analysis.

From the above discussion, it becomes apparent that so far the related work on the topic presents two major shortcomings which are dealt by the work at hand:

1. It does not consider more sophisticated attacks scenarios like DDoS and low-rate DoS, which naturally are more difficult to detect and defeat. The works in [10,30] do take into consideration the aforementioned category of assaults but either the reported results are only concerned with offline detection or they solely address elementary DoS attacks.
2. No work occupied with real-time detection comprehensively reports on the overhead that the various classifiers induce to the system in terms of classification time. However, this is crucial for deciding if a particular detection method is suitable or not.

6 Conclusions

This work examines the applicability of several well-known ML techniques in detecting (D)DoS in SIP-based services. We implemented a Kamailio software module to achieve attack detection in realtime and tested both the detection accuracy and processing overhead of each classifier under 15 different attack scenarios representing DoS and DDoS incidents. All the algorithms but one achieve desirable detection accuracy in terms of FP, but only mediocre accuracy (for a real IDS product) when FN is considered. Therefore, further experimentation is needed to better appreciate this potential. From a processing overhead viewpoint, it can be safely argued that realtime operation is feasible as the induced time penalty is negligible even if the SIP server operates in single-thread mode.

It is worthy to note that the legitimate users' calling rate, the number of users included in every scenario, and the value of the M_w are sure to affect the false alarm rates. This happens because the selected features rely on the occurrences of the examined headers of interest, and thus are directly connected to the number of users and the M_w. That is, the greater the number of users the lesser the occurrences per message header. This fact indicates that one should lower the size of M_w as the population of users augments and vice versa. To this end, one can obtain the best results with an error-trial approach, calibrating the M_w proportionally to the population of users and the calling rate as the case may be.

Acknowledgements

This paper is part of the 5179 (SCYPE) research project, implemented within the context of the Greek Ministry of Development-General Secretariat of Research and Technology funded program “Excellence II / Aristeia II”, co-financed by the European Union/European Social Fund - Operational program “Education and Life-long Learning” and National funds.

References

1. C. Mohr, “Report: Global voip services market to reach 137 billion by 2020,” Nov. 2014. [Online]. Available: http://www.tmcnet.com/channels/hosted-softswitch/articles/393593-report-global-voip-services-market-reach-137-billion.htm.
2. D. Geneiatakis, G. Kambourakis, C. Lambrinoudakis, T. Dagiuklas, and S. Gritzalis, “Sip message tampering: The sql code injection attack,” in *Proceedings of 13th International Conference on Software, Telecommunications and Computer Networks (SoftCOM 2005), Split, Croatia*, 2005.
3. D. Geneiatakis, T. Dagiuklas, G. Kambourakis, C. Lambrinoudakis, S. Gritzalis, K. Ehlert, and D. Sisalem, “Survey of security vulnerabilities in session initiation protocol,” *Communications Surveys Tutorials, IEEE*, vol. 8, no. 3, pp. 68–81, rd 2006.
4. D. Geneiatakis, G. Kambourakis, C. Lambrinoudakis, T. Dagiuklas, and S. Gritzalis, “A framework for protecting a sip-based infrastructure against malformed message attacks,” *Communications Networks, Elsevier*, vol. 51, no. 10, pp. 2580–2593, 2007.
5. G. Kambourakis, C. Kolias, S. Gritzalis, and J. H. Park, “Dos attacks exploiting signaling in UMTS and IMS,” *Computer Communications*, vol. 34, no. 3, pp. 226 – 235, 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S014036641000085X.
6. M. Shtern, R. Sandel, M. Litoiu, C. Bachalo, and V. Theodorou, “Towards mitigation of low and slow application ddos attacks,” in *Cloud Engineering (IC2E), 2014 IEEE International Conference on*, March 2014, pp. 604–609.
7. S. Ehlert, G. Zhang, D. Geneiatakis, G. Kambourakis, T. Dagiuklas, J. Markl, and D. Sisalem, “Two layer denial of service prevention on sip voip infrastructures,” *Computer Communications*, vol. 31, no. 10, pp. 2443–2456, 2008.
8. Z. Tsiatsikas, D. Geneiatakis, G. Kambourakis, and A. D. Keromytis, “An efficient and easily deployable method for dealing with dos in sip services,” *Computer Communications*, vol. 57, no. 0, pp. 50 – 63, 2015.
9. S. Ehlert, D. Geneiatakis, and T. Magedanz, “Survey of network security systems to counter sip-based denial-of-service attacks,” *Computers & Security*, vol. 29, no. 2, pp. 225–243, 2010.
10. Z. Tsiatsikas, A. Fakis, D. Papamartzivanos, D. Geneiatakis, G. Kambourakis, and C. Kolias, “Battling against ddos in sip - is machine learning-based detection an effective weapon?” in *Proceedings of the 12th International Conference on Security and Cryptography*, 2015, pp. 301–308.
11. J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, “Sip: Session initiation protocol,” Internet Requests for Comments, June 2002. [Online]. Available: http://www.rfc-editor.org/rfc/rfc3261.txt.
12. A. D. Keromytis, “A comprehensive survey of voice over ip security research,” *IEEE Communications Surveys and Tutorials*, vol. 14, no. 2, pp. 514–537, 2012.
13. SIPVicious. (2016) Sipvicious. http://blog.sipvicious.org/.

14. C. S. Advisory, “Cisco sip phone 3905 resource limitation denial of service vulnerability,” 2015. [Online]. Available: https://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-20151202-sip.
15. Z. Tsiatsikas, D. Geneiatakis, and G. Kambourakis, “Research project scype: Software modules.” [Online]. Available: https://scype.samos.aegean.gr/tzisis/scype_5179_software/.
16. Kamailio. (2014) the open source sip server. http://www.kamailio.org/w/.
17. M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, “The weka data mining software: An update,” *SIGKDD Explor. Newsl.*, vol. 11, no. 1, pp. 10–18, Nov. 2009.
18. R. Gordon, “Essential jni: Java native interface.” Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1998.
19. Oracle, “Crashing jvm,” 2016. [Online]. Available: http://docs.oracle.com/cd/E15289_01/doc.40/e15059/crash.htm#i1010768.
20. SIPp, “Free open source test tool / traffic generator for the sip protocol.” [Online]. Available: http://sipp.sourceforge.net/index.html.
21. N. Ohlmeier, “Sip swiss army knife.” [Online]. Available: http://sipsak.org/.
22. J. Stanek and L. Kencl, “Sipp-dd: Sip ddos flood-attack simulation tool,” in *Computer Communications and Networks (ICCCN), 2011 Proceedings of 20th International Conference on*, July 2011, pp. 1–7.
23. R. Krishnamurthy and G. Rouskas, “Evaluation of sip proxy server performance: Packet-level measurements and queuing model,” in *Communications (ICC), 2013 IEEE International Conference on*, June 2013, pp. 2326–2330.
24. I. H. Witten, E. Frank, and M. A. Hall, *Data Mining: Practical Machine Learning Tools and Techniques*, 3rd ed. Burlington, MA: Morgan Kaufmann, 2011. [Online]. Available: http://www.sciencedirect.com/science/book/9780123748560
25. C. E. Shannon, “A mathematical theory of communication,” *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 5, no. 1, pp. 3–55, Jan. 2001.
26. M. Nikulin, *Hellinger distance.* Encyclopeadia of Mathematics, 2001.
27. Y. Bouzida and C. Mangin, “A framework for detecting anomalies in voip networks,” in *Availability, Reliability and Security, 2008. ARES 08. Third International Conference on.* IEEE, 2008, pp. 204–211.
28. M. A. Akbar and M. Farooq, “Application of evolutionary algorithms in detection of sip based flooding attacks,” in *Proceedings of the 11th Annual conference on Genetic and evolutionary computation.* ACM, 2009, pp. 1419–1426.
29. M. Nassar, O. Festor *et al.*, “Monitoring sip traffic using support vector machines,” in *Recent Advances in Intrusion Detection.* Springer, 2008, pp. 311–330.
30. M. A. Akbar and M. Farooq, “Securing sip-based voip infrastructure against flooding attacks and spam over ip telephony,” *Knowledge and information systems*, vol. 38, no. 2, pp. 491–510, 2014.
31. M. Z. Rafique, Z. S. Khan, M. K. Khan, and K. Alghatbar, “Securing ip-multimedia subsystem (ims) against anomalous message exploits by using machine learning algorithms,” in *Information Technology: New Generations (ITNG), 2011 Eighth International Conference on.* IEEE, 2011, pp. 559–563.