Development and Analysis of Generic VoIP Attack Sequences Based on Analysis of Real Attack Traffic

Adnan Aziz, Dirk Hoffstadt, Sebastian Ganz, Erwin Rathgeb
Computer Networking Technology Group
University of Duisburg-Essen
Essen, Germany
adnan.aziz@uni-due.de, {dirk.hoffstadt, erwin.rathgeb}@iem.uni-due.de, sebastian.ganz@stud.uni-due.de

***Abstract* - Security issues like service misuse and fraud are emerging problems of SIP-based networks. To devise effective countermeasures it is important to know how these attacks are launched in reality. Multi-stage attacks to commit *Toll Fraud* are already known in principle. We have identified different variations in these attack patterns by analyzing over 25 GByte of SIP attack traffic collected in our SIP Honeynet over a period of three years i.e., from December 2009 to November 2012. Based on this analysis, we have developed a Generic Attack Replay tool (GART) which allows replaying samples of the major attack variants in arbitrary network setups. This tool can be used for evaluation of detection and mitigation components where realistic and reproducible attack traffic is needed. The tool described here and the sample database will be made available to interested groups.**

Keywords: SIP; STR; analyze; misuse; fraud; security; VoIP; attack patterns; Honeynet; evaluation

I. INTRODUCTION

Voice-over-IP (VoIP) communication based on the Session Initiation Protocol (SIP) [1] has evolved as a de-facto standard for voice communication. Therefore, support of open SIP-based interfaces is an increasingly important requirement for IP-based Public Branch eXchanges (PBXs) and provider systems. This, however, opens up new opportunities for misuse and fraud. Remote access to SIP server enables fraudulent registration attempts (known as Registration Hijacking) as a prerequisite for calls via compromised SIP accounts. This is particularly attractive for attackers, because they can gain immediate financial benefits by making toll calls (international, cellular, premium calls) via third party accounts. This is called Toll Fraud and can cause the account owner substantial financial damage in a very short time.

In order to observe and analyze SIP-based attacks, we have developed a specific SIP Honeynet system [2], which operates on the Internet since December 2009 and has collected over 90 million SIP messages in total until January 2013. One main component of the system is the SIP Trace Recorder (STR) [3], allowing monitoring and recording of all SIP traffic in complete subnets. The system also provides automatic correlation of SIP messages in order to allocate them to distinct attack incidents. This allows us to analyze the various attack patterns in detail and to identify typical variations.

Based on this analysis, we have generated a database with currently 5684 attack samples representing the major attack variants. The Generic Attack Replay Tool (GART) presented in section VI allows to adapt these attack samples to a specific network setup (e.g., IP address of the attack target) and to replay the sample attacks preserving all their characteristics. Thus, the GART can be used to reproduce a broad range of typical SIP attack patterns, which is crucial for evaluation and calibration of SIP specific detection and mitigation algorithms and components. The sample database will be updated periodically and made available to interested groups.

The remainder of this paper is organized as follows: Section II gives a short overview of SIP and different stages of typical multi-stage attacks. An overview of the STR is given in section III. The related work is covered in section IV, followed by the different attack sequences identified in each attack stage in section V. Section VI presents the GART tool, developed to replay the real-time attack traffic. Finally, the section VII concludes the paper.

II. VOIP-SPECIFIC MISUSE

SIP is used to establish sessions (e.g., voice, video) between two user agents. A user interacts with SIP via an entity called user-agent. It contains an interface towards the user e.g., a software installed on a computer. For the purpose of this paper, the following message types are relevant: If a user agent (i.e., SIP device) wants to establish a call via a voice server in SIP-based networks, a registration at the server is necessary. In order to register, a user agent sends a REGISTER message with credentials (account name and password) to the server. If the extension (SIP account) given in the REGISTER message exists and the password is correct, the server acknowledges with a 200 OK message. Else, the SIP server either responds with a 401 UNAUTHORISED message if the password is empty, with a 403 FORBIDDEN message if the password is incorrect or with a 404 NOT FOUND message if the account does not exist. The numeric part, e.g., 200 is called response code and the string part, e.g., OK, is called response phrase. OPTIONS messages allow a user agent to query a server's capabilities and to discover information about the supported SIP methods, extensions, codecs, etc. without establishing a session. To ensure that this communication is always possible, the standard specifies that an OPTIONS packet must be answered, regardless of its source or existing connections.

To better understand the work explained in this paper some SIP header fields are important to understand. Consider the SIP header shown in Figure 1 for information about these fields.

The first row in Figure 1 is called the request line. It contains the SIP method used, the Uniform Resource Identifier (URI) of the target and version of SIP protocol used. The “via” header is the local address of the caller, i.e. the address where he is expecting the SIP server response. The “to” header field specifies the desired logical recipient of the request. It consists of display name (Bob Johnson) and the SIP URI. The part of “to” header’s SIP URI in front of the “@” sign is called to-user whereas the part behind it, i.e., “server2.com”, is called to-host. The “from” header field serves as an identifier of the caller. It contains the display name of the caller followed by his SIP URI. The user part of the “from” header is called from-user, whereas “server1.com” is called from-host. The call-ID field, a pseudo random number, is used to uniquely identify a call. The “contact” header field contains a SIP URI that represents a direct route to the caller. It contains a username called contact-user and an IP address which is called contact-host.

To finally exploit a third party SIP extension, typically four distinct attack stages are necessary which we have observed during our Honeynet field test [2].

1. SIP Server & Device Scan

The SIP protocol requires every SIP device to answer OPTIONS packets. An attacker can use this behavior to “ping” any single IP address or whole subnets with OPTIONS packets to identify SIP devices. Even if a user agent’s SIP stack implementation is not standard compliant and replies only to OPTIONS packets of well-known sources, a scan is also possible: In this case, the attacker can use REGISTER requests instead of OPTIONS messages.

2. Extension Scan

To identify active extensions of known SIP servers, the attacker tries to register at several extensions without using a password. An extension identifier consists of digit sequences and/or strings. If the extension exists, the server normally answers with a 403 FORBIDDEN, because no password is given.

```
INVITE sip:bob@server2.com SIP/2.0
via: SIP/2.0/UDP pc3.server1.com;   branch=z9hG4bK776asdhds Max-
     Forwards: 70
to: Bob Johnson<sip:bob@server2.com>
from: Alice White <sip:alice@server1.com>;tag=1928301774
call-ID: a84b4c76e66710@pc3.server1.com
cSeq: 131495 INVITE
contact: <sip:alice@pc3.server1.com>
content-Type: application/sdp
content-Length: 142
```

Figure 1: SIP Header

If it does not exist, a 404 NOT FOUND is typically returned. The result of this attack stage is a complete list of existing extensions (provider accounts).

3. Registration Hijacking

To register at a given extension, the attacker tries to guess the password. This results in sending a sequence of – possibly very many – REGISTER messages with different passwords to a selected extension. If the password is guessed, the information is stored to register at this extension later on.

4. Toll Fraud

The term “Toll Fraud” is used if a person generates costs (toll) by misusing the extension of another person. In this case, an attacker has already successfully hijacked an extension and uses the VoIP functionality to make calls, specifically international calls or calls to premium numbers. Another motivation to use a hijacked account for a call is to obfuscate the caller identity. In terms of SIP messages, the attacker first sends a REGISTER message with the correct password. After the 200 OK message from the server, the attacker can initiate calls by using INVITE messages.

The first three stages mentioned above can be executed by using freely available tool suits. A common white-hat attacking tool for SIP is the open source tool suit Sipvicious [4]. It contains several small programs: The first one is a SIP scanner called “svmap”. It scans an IP address range for SIP devices, either sequentially or in a random order, typically with OPTIONS packets. Sipvicious also provides tools to find active SIP accounts with REGISTER messages (“svwar”) and to crack passwords (“svcrack”). If not modified, Sipvicious identifies itself as user agent “friendly-scanner”.

III. SIP Trace Recorder (STR)

The SIP Trace Recorder (STR) [3] is a tool developed by our group. As shown in Figure 2, the STR is connected to the monitoring port of a layer 3 switch in our Honeynet setup. It is used to monitor a subnet with globally reachable virtual Honeypots [2] that interact with the attacker. The fact that public IP addresses are used for the Honeynet system ensures that an attacker can easily reach the Honeynet system via the Internet. The captured SIP traffic is queued, a timestamp is generated and the header values, e.g., source and destination IP, SIP method, user-agent, etc. are analyzed. This information is then stored into an SQL database called STR database. The STR allows observing whole subnets. Its analysis modules contain several plugins consisting of SQL queries and PHP scripts to provide powerful analysis and graphical representation. To better understand the attack attempts, the STR clusters attack messages based on various parameters, e.g., message types, source IP address and timestamp. It also generates reports for attack stages, e.g., for each attack stage, the number of SIP requests and corresponding number of attacks are presented in tabular format on daily basis.

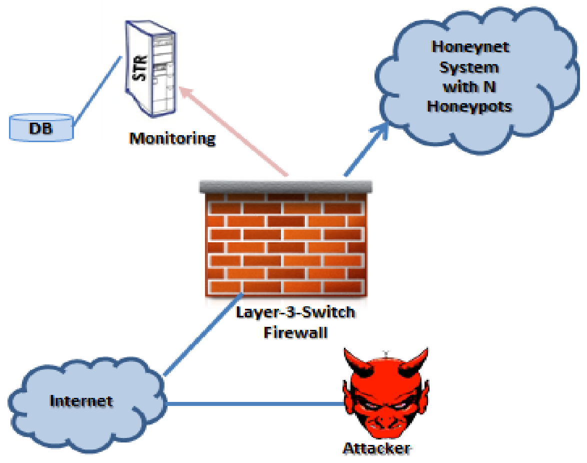


Figure 2: STR setup

An event correlation for Toll Fraud calls is automatically performed. If a call is established via our Honeynet system, the STR generates Call Data Records (CDR) by correlating the SIP INVITE and BYE messages.

The STR can be used in different environments and locations by combining it with other components or plugins. The STR also provides a privacy option. Therefore, it can be used in both passive and productive environments.

IV. RELATED WORK

In [7], different SIP Denial of Service (DoS) attacks are analyzed. Three algorithms are defined to detect and mitigate DoS and Distributed DoS (DDoS) attacks. To differentiate the different variations from regular traffic, a specification based detection scheme is introduced. This work provides a promising solution towards SIP DoS attacks. However, in order to identify attacks other than DoS attacks on SIP based networks it is necessary to monitor and analyze the SIP traffic in more detail.

The Honeypot concept has been introduced to detect unauthorized access to information systems [8]. Different variants of Honeypots were introduced, each used for a different purpose. Low Interaction Honeypots only simulate the existence of just one service to a limited extent and log access to it. Whereas, High Interaction Honeypots are based on real service implementations and operating systems, providing full functionality. A Honeynet is a network consisting of one or more Honeypots. A Honeypot appears as a part of the network and as a real productive system but it is separated and strictly monitored so that it may not pose a threat to the productive system.

In [9], an Intrusion Detection System (IDS) has been built to detect SIP attacks. This system is based on the Low Interaction Honeypot presented in [10]. The IDS detects DoS and Call attacks by working with a security event correlation system. The Honeypot is capable of retrieving fingerprint information by interacting with the attacker. However, this approach is limited to DoS attack detection and Call-based attacks.

In [11], Valli has performed a statistical analysis of VoIP attacks over the real attack traffic captured via a Honeynet system consisting of several virtualized Low Interaction Honeypots. These Honeypots have logged the target traffic to a file, over which an analysis was performed. The results have shown that primarily Sipvicious is used as a tool. Also another tool called sipsscuser is found. Its behavior is found similar to a worm or a botnet.

In the above mentioned work, data at specific Honeypot is captured and a packet level analysis is performed. However, to identify all the attack patterns, it is necessary to analyze a complete network and to perform a stateful analysis. In [1] our SIP Honeynet system has been introduced. It allows bi-directional packet correlation for stateful analysis. The system is extended by a rule based clustering algorithm to allocate multiple attack packets to individual attack stages. It provides a much clearer view on attack behaviors.

In the following sections we analyze the mentioned attack stages more in detail and identify different attack variations and patterns.

V. ANALYSIS OF ATTACK PATTERN VARIATIONS

The SIP Trace Recorder (STR) [3] determines the attack stages (SIP Device and Server Scan, Extension Scan, Registration Hijacking and Toll Fraud) by clustering related SIP messages. These message clusters need to be further analyzed to find out the typical patterns related to these attack stages. We have thoroughly analyzed the STR database [3] containing attack traffic collected until November 2012. The analysis has shown that each of these attack stages shows several variations.

1. SIP Server & Device Scan Variations

In a SIP Server Scan, attackers typically do not sequentially scan all IP addresses in a range, but randomly choose the next address to scan. This is exactly the standard behavior of the attack tool Sipvicious. An attacker sends OPTIONS packets (typically 32) to one IP address and then disappears regardless of the response. After a substantial period of time, typically several hours, a scan of another IP address is performed using the same source IP as before.

To identify the variations in Device and Server Scan, the OPTIONS messages were extracted from the STR database. Grouping this data by contact, user-agent and to header fields, multiple variations were identified. Some of them are listed in Table 1. The most significant differences among variations are the timing between attack packets, contact header and fingerprints for different attack tools.

Variation SS-a uses a class C private IP address as contact-host for every OPTIONS message sent. This is currently the only variation associated with the user-agent sundayddr. On average approximately 12,833 packets are sent per distinct source IP address indicating a thorough scanning. The number of Server Scan attacks recorded belonging to variation SS-a were 72. In the “to” and “from” parts of user and host two

TABLE 1: Server Scan Variations

Var.	SS-a	SS-b	SS-c	SS-d	SS-e	SS-f	SS-g
contact-host	192.168.1.9	source-IP	Random class A, B or C IP	Unrelated IP to source-IP and destination-IP, proxy server, “ “	Loopback address	source-IP	source-IP
user-agent	sundayddr	eyebeam release 3006o stamp 17751, sipv, Trixbox, friendly-scanner	eyebeam release 3006o stamp 17751, friendly-scanner, friendly-request	Friendly-scanner / ” ”	Friendly-scanner / Asterisk PBX	Friendly-scanner / Asterisk PBX	friendly-scanner
to-host	192.168.1.9 / “ ”	1.1.1.1	1.1.1.1	destination IP / 1.1.1.1 / nm2	unknown IP/ 1.1.1.1 / “ “	unknown IP / “ “	destination-IP
from-host	192.168.1.9 / “ ”	1.1.1.1	1.1.1.1	destination IP / 1.1.1.1/ nm	unknown IP/ 1.1.1.1 / loopback add / “ “	unknown IP / source-IP	destination-IP
contact-user	100	100	100	100	100 /1000 / b	100 / a	100
to-user	100 / “ ”	100	100	default / 100 / nm2	100	100 / “ “	default
from-user	100 / “ ”	100	100	default / 100 / nm2	100 / b	100 / a / “ ”	default
# of pkts	927611	923,616	90,786	63,067	543042	17669	163
# of attacks	72	527	79	136	959	203	2
Avg. pkts per src IP	~12,883	~1,753	~1,149	~464	~566	~87	~82

values were identified. One of them is an empty string, which might be because the tool failed to set it or an attacker has set these values as empty.

Unlike variation SS-a, variation SS-b uses the source-IP address as contact-host entry. It also uses different user-agent names. Contrary to variation SS-a, the “to” and “from” host parts of the URI are of type 1.1.1.1, whereas the “to” and “from” user parts are always 100 as in variation SS-a. On average, about 1,753 packets are sent per source IP address. This scanning is not as thorough but can also be quickly detected as in SS-a, because a significant number of messages arrives at the server within a second or less. This variation has been observed quite often, in total 527 times. An analysis of the implementation of Sipvicious we use in our lab has shown that the variation SS-b is performed using “standard” Sipvicious as the signatures of this variation match exactly the behavior of the Sipvicious tool.

A header level analysis of the Sipvicious and sundayddr scanners shows that in both tools the order and layout of the header fields is very similar – which indicates that the sundayddr scanner is an adaptation of Sipvicious.

The variation SS-c is similar to variation SS-b but it uses random class A, B and C IP addresses as contact-host entry. About 4% of Server Scan attacks belong to variation SS-c. In variations SS-b and SS-c, the user-agent field can contain the string “eyebeam” that indicates a popular softphone. However, it is not feasible to manually generate scans with the timing observed, and the number of OPTIONS messages eyeBeam normally sends is much less than the values in the Table 1. The other header values for these attack variations match the behavior of Sipvicious and sundayddr. Also the eyeBeam version found is the same for all attacks. These are indications that the string is set by the attacker when using one of these tools to somehow camouflage the attack.

Variation SS-d uses dissimilar source and destination IP addresses as contact-host header value. Friendly-scanner is identified as user-agent for this variation. In this case, the attack behavior is quite different form a typical softphone and invalid SIP messages are sent. The Server Scan attacks identified by using variation SS-d are roughly double that of variation SS-c.

Almost 50% of the total Server Scan attacks identified were performed using variation SS-e. It uses the loopback address as contact-host header value and friendly-scanner & Asterisk PBX as user-agent. Therefore, a SIP server is not able to establish a session due to the fact that an invalid contact address is used by the attacker. Variation SS-f occurs 203 times (10% of the total Server Scan attacks) and is similar to variation SS-e, however, it uses the source IP as contact-host. Thus, in contrast to SS-e, a SIP session could be established. Variation SS-g is alike variation SS-b, however, the number of scans performed using this variation is only 0.1% of the total Server Scans and much less than variation SS-b. It is hard to identify the variations SS-f and SS-g as the number of messages per scan is very small.

Comparing the different header fields presented in the Table 1, we can see that the signatures of variations SS-c, SS-d, SS-e and SS-f are similar to the signatures of the Sipvicious tool, represented by variation SS-b, with some variations. These variations cannot be generated by simply configuring Sipvicious accordingly but require changes in the Sipvicious Python script. This indicates that attackers are starting to create modifications of Sipvicious and sundayddr to avoid simple detection based on existing signatures.

2. Extension Scan Variations

To probe for correct account extensions two major techniques have been observed: One is a simple probing of numbers, probably generated by a “for” loop or random number generator. The other is a kind of dictionary attack, where typical names, e.g., “default”, are probed.

To analyze the Extension Scan stage, relevant REGISTER messages from the STR database [3] were taken and grouped by user-agent, contact-host, to-host and from-host fields. Comparing contact-user and to-user header we have classified two variations, ES-a and ES-b. The Variation ES-a represents those Extension Scan attacks where contact-user and to-user fields are not equal whereas the variation ES-b comprises Extension Scan attacks where contact-user and to-user fields are alike. Variation ES-b is subdivided into four subclasses as shown in Table 2. It shows that Extension Scans are currently exclusively performed using friendly-scanner as user-agent. Variations ES-b3 and ES-b4 contain values 3 and k respectively for contact-host, to-host and from-host header fields. These values are obviously set by the attacker to disguise the identity.

Comparing the Extension Scan variation signatures with the standard implementations of Sipvicious and the eyeBeam softphone, we have observed that attack signatures (contact-user, contact-host, to-host and from-host header values) of variation ES-b1 are exactly the same as found in the Sipvicious tool. The signatures of variation ES-b2 are the same as the behavior of eyeBeam. Therefore, only in variation ES-b2 a valid registration request is used as expected from a real softphone request. This analysis again indicates that attackers have started to modify the standard tool implementations to optimize and camouflage the attacks.

The variations identified in Table 2 were also categorized on the basis of the number of REGISTER requests sent. Table 3 shows this analysis. It shows that almost 48% of the attacks in variation ES-a have sent less than 1000 packets to perform an Extension Scan and almost 43% Extension Scans are performed with packets ranging between 1000 and 10000. Variations ES-b1 and ES-b2 perform very thorough scans based on dictionaries of varying sizes whereas variations ES-b3 and ES-b4 perform limited scans using “for” loops or random numbers.

TABLE 2: Extension Scan Variations

Variation	**user-agent**	**contact-user**	**contact-host**	**to-host**	**from-host**	**# of scans**
ES-a	friendly-scanner	123	1.1.1.1	destination IP	destination IP	23
ES-b1	friendly-scanner	to-user	destination IP	destination IP	destination IP	50
ES-b2	friendly-scanner	to-user	source IP	destination IP	destination IP	4
ES-b3	friendly-scanner	to-user	3	3	3	13
ES-b4	friendly-scanner	to-user	k	k	k	3

TABLE 3: Subclasses Specified by # of Scanned Extensions

variation \ # of packets	**< 1000**	**1000 – 10000**	**10000 – 100000**	**>= 100000**
ES-a	11	10	1	1
ES-b1	6	5	35	4
ES-b2	0	0	4	0
ES-b3	13	0	0	0
ES-b4	3	0	0	0

3. Registration Hijacking Variations

An interesting fact has been observed that the user agent “sundayddr” does not try to crack accounts. Not even a single “REGISTER” message of this agent is visible in the traces. We identified two types of Registration Hijacking which can be differentiated by analyzing the scanning technique: If the number of SIP messages per attack is enormous (over one million), the attacker uses a dictionary attack to crack the Honeypot SIP account. In case of a lower packet count, the attacker only uses iterative or random numbers.

To identify the variations of the Registration Hijacking scans, we have grouped the Registration Hijacking scan data from the STR database [3] on the basis of user-agent, contact-user, contact-host, to-host and from-host header fields. We have identified two variants of Registration Hijacking scan as shown in Table 4.

The difference in the identified variations is the contact-host header field. Most of the scans use an IP address which is invalid for this registration request in the contact-host field whereas very few scans are using the valid source IP address. The header level analysis indicates that the variations RH-a and RH-b are performed by Sipvicious. The standard implementation of Sipvicious sets the to-user header field to “123” and the contact-host to “1.1.1.1”. In the variation RH-b, however, the attack tool mimics a valid softphone behavior more closely, which confirms that attackers are using some modified versions of these tools.

On the basis of the number of passwords tried to hijack an account, variations RH-a and RH-b are further subdivided into subclasses as shown in the Table 5. It shows that most of the attacks have used iterative or random numbers for Registration Hijacking scans.

TABLE 4: Registration Hijacking Variations

Variation	**user-agent**	**contact-user**	**contact-host**	**to-host**	**from-host**	**# of scans**
RH-a	friendly-scanner	123	1.1.1.1	destination IP	destination IP	302
RH-b	friendly-scanner	123	source IP	destination IP	destination IP	18

TABLE 5: Subclasses Specified by # of Passwords Used

variation	# of tried passwords	# of scans
RH-a1	< 1,000	124
RH-a2	1,000 - 10,000	60
RH-a3	10,000 - 100,000	105
RH-a4	>= 100,000	13
RH-b1	< 1,000	5
RH-b2	1,000 - 10,000	0
RH-b3	10,000 - 100,000	12
RH-b4	>= 100,000	1

Only a few scans have used massive dictionary attacks for Registration Hijacking. This can, at least partly, be attributed to the fact that dictionary attacks require more effort by the attacker to get or compile a dictionary as this is not part of the Sipvicious tool, whereas the scan using iterative or random numbers is already integrated.

4. Toll Fraud Variations

It is very difficult to distinguish between a call from a genuine user and an attacker in a productive environment. In our Honeypot setup, however, all traffic is per definition attack traffic since we have no legitimate users. The Honeypot setup does not establish calls towards the target addresses. Therefore, the attacker always observes a failure in call establishment and eventually loses interest in that specific account. Before he does, we can still observe some specific behavior.

Since the attacker is not aware of the local configuration of even the country where the server is located, he has to test various dialing prefixes in order to get out of the local "SIP PBX" and eventually also out of the national network – this is behavior which would also be observed in a productive environment. However, in our setup the attacker can never be successful.

Common softphones send multiple INVITE messages for initiating a call, e.g. eyeBeam typically three (e.g., the first request without credentials which results in an UNAUTHORIZED response message from the SIP server). Since we are not interested in single, manual call attempts, we have set a threshold on the number of INVITE packets required to identify an attack. If more than 5 INVITE requests are detected from a user in a period of one second, such activities are considered as automated Toll Fraud attack. To identify the variations in these Toll Fraud attacks, we have extracted Toll Fraud attack data from STR database and grouped it on the basis of source IP, source-port, destination IP and time. Table 6 shows the different variations identified.

Three variations were found. Variation TF-a uses a random hexadecimal string concatenated by "@" sign and source IP. The open source PBX "Asterisk" [6] indicated as user client can be used to automatically perform Toll Fraud attacks – and specifically the necessary probing described earlier – due to the fact that Asterisk provides manager and "call file" interfaces.

TABLE 6: Toll Fraud Variations

Variation	call-id	user-agent	contact-user	# of scans
TF-a	"string"@sourceIP	Asterisk PBX	from-user	854
TF-b	"string"@destinationIP	Asterisk	from-user	6
TF-c	random character sequence	22 different	from-user	2,433

Variation TF-b also uses a random hexadecimal string but concatenated with "@" and destination IP. The small number of scans indicates that Toll Fraud is performed manually using Asterisk with different configurations. Variation TF-c uses a random character sequence as Call-ID. Contrary to TF-a and TF-b, 22 different user-agents have been identified in variation TF-c to perform Toll Fraud. The large number of scans performed indicates that script-controlled softphones or the Asterisk PBX manager or "call file" interfaces are used for this variation.

The analysis described above provides a categorization of typical attack patterns. The analysis confirms that the attacks are performed primarily by using Sipvicious. However, some of the observed variants require modifications in the source code of the tool. This indicates that attackers are optimizing the tool in order to mimic the behavior of a real softphone more closely to avoid a simple discovery of the attacks.

Another result is a set of 5684 attack samples for the different attack stages sorted into the classes introduced. All SIP messages related to the samples are available in full detail including the timing information between related messages. This set of attack samples can be used, e.g., to test and calibrate comprehensive detection and mitigation algorithms and components under realistic conditions as shown in the next section.

VI. GENERIC ATTACK REPLAY TOOL (GART)

In order to make this data practically available to test detection and mitigation components in a local network, a Generic Attack Replay Tool (GART) has been developed. It is implemented in Java using SQLite database [12] and JAIN SIP [13] (Java API). This allows the usage of GART on multiple platforms. Our SIP Honeynet system [2] is running since December 2009 and the STR database [3] contains over 90 million SIP messages in the meantime. To obtain a current but comprehensive set of attack samples, we have created a SQLite database file from the STR database recorded over a period of three years i.e., from December 2009 to November 2012. The SQLite database [12], a small subset of the complete STR database, contains four tables which were created with the help of different SQL queries. It contains representative samples of the variations identified for Server Scan, Extension Scan, Registration Hijacking and Toll Fraud in a tabular format as shown in Figure 3.

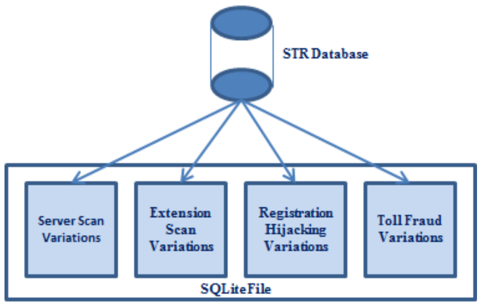


Figure 3: SQLite Database

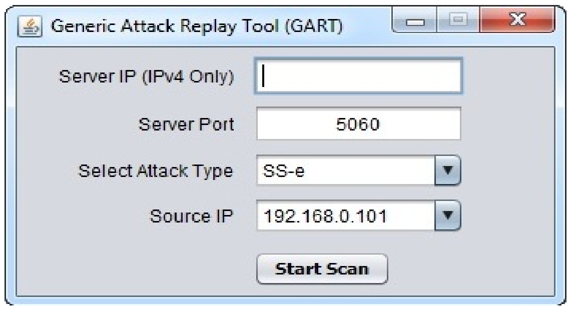
To make it simple and lightweight, the SQLite database currently contains only one example per variation. However, it can be easily extended to contain various specific subsets of attack samples.

To make GART simple and easy to use, a GUI has been created to control the replay of attack traffic. As shown in Figure 4, it takes server IP, server port, attack type and source computer IP address as inputs from the user in order to replay the attack traffic in real-time.

The Server IP is the IP address of the SIP server we want to scan and the Server port is the port where the SIP server listens to requests. The attack type contains the list of variations identified in section V. A dropdown list is provided which is populated from the SQLite database. The user simply selects the variation to be replayed from this list. The last input into the GUI is the automatically detected IP address of the user agent client, where server responses will be received by GART.

When the *Start Scan* button is pressed, multiple actions are performed, resulting in two log files. These log files contain logs of SIP server responses for the requests sent by GART during the replay scan. For good overview and quick check, the first log file is kept short and contains only the timestamp, status-code and reason phrase of the SIP server response, whereas the second log file contains the complete SIP server response messages. These log files can be used for offline analysis and to find out if the tested target hosts send the expected responses or not.

The attack packets in the database have been recorded in our specific network environment. To successfully send messages to the attacked SIP server in an arbitrary test network and receive the server responses, some changes have to be made in the header fields.



Server IP (IPv4 Only)	
Server Port	5060
Select Attack Type	SS-e
Source IP	192.168.0.101

Figure 4: GART GUI

Therefore, GART automatically adapts SIP header values like source IP, destination IP, destination-port and via. However, the time interval between messages is kept unchanged to replay the attack traffic in the local setup preserving the major characteristics of attack.

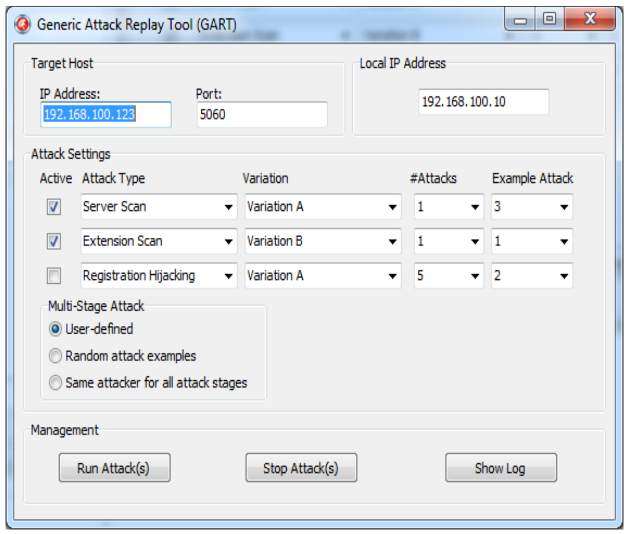
In order to perform SIP Device and Server Scan, OPTIONS messages from SQLite database are sent to a class C network. To map real attack traffic to the local network and to scan complete subnets, the first three octets of the specified Server IP from the GUI are concatenated with the last octet of the destination IPs stored in SQLite database. Other header values are taken from the database to complete the packet.

To perform an Extension Scan, Registration Hijacking scan or Toll Fraud attack, the Server IP address input by the user serves as address of the SIP server and other header values are modified by the tool according to the local network. Completion of a scan is indicated by a message and the tool is ready for a new scan.

A functional test was performed to evaluate that the SIP requests are correctly generated. The test environment consisted of a 64-bit Ubuntu machine running on a virtual machine, executing GART and Wireshark [5] plus four Asterisk [6] servers running in a class C network. Wireshark captured the SIP traffic between the virtual machine network interface and the Asterisk servers.

Five scans for each variation type were executed. After each scan, the Wireshark trace file was analyzed to check the consistency of sent packets and expected response messages. The server responses received have shown that all of the packets were successfully delivered.

The initial GART version was capable of replaying a particular example of a specific attack type. Currently, it is extended to perform more actions as shown in the Figure 5. It will allow the user to customize the attacks to be replayed on an arbitrary network. The user will be able to select whether to replay a complete attack with all attack stages involved or to replay some particular attack stages only.



Target Host
IP Address: 192.168.100.123
Port: 5060
Local IP Address: 192.168.100.10

Attack Settings

Active	Attack Type	Variation	#Attacks	Example Attack
✓	Server Scan	Variation A	1	3
✓	Extension Scan	Variation B	1	1
	Registration Hijacking	Variation A	5	2

Multi-Stage Attack
- ● User-defined
- ○ Random attack examples
- ○ Same attacker for all attack stages

Management

Figure 5: Extended GART GUI

The user will also have the option to choose the variation type per attack stage selected and also to select the number of attacks. Due to reproducible tests, the user is able to select a specific example attack per variation. The extended version of GART will help us to improve the performance of our detection and mitigation components.

VII. CONCLUSIONS

In this paper, we have presented an analysis of real-life SIP attacks and identified typical variations in the various stages of this multi-stage attack. The attack traffic was taken from a comprehensive attack database, observed over a long period of time. Different modifications of attack tools, e.g., Sipvicious have been identified. The signatures of a softphone, eyeBeam, have been observed in different attack stages, purposely used by the attackers to mislead the detection and mitigation components. This is a clear indication that attackers optimize and customize the available attack tools.

The attack data stored in our database preserves all relevant characteristics of original attacks including the timing which makes it useable to conduct consistent and reproducible tests. On the basis of the analysis and the identified variations, a Generic Attack Replay Tool (GART) has been developed. It allows the use of attack samples in an arbitrary network setup with minimum configuration effort. It is used to replay the sampled real attacks into an arbitrary network by preserving all the characteristics of the attack data. It also provides a broad coverage of attack patterns as it contains one sample per variation detected. It, therefore, can be used to efficiently reproduce attack patterns to evaluate and calibrate SIP-based detection and mitigation algorithms and components. The sample database and the tool are maintained and extended and will be made available to interested groups upon request.

Currently we are using the results to generate attack signatures based on the variations analysis and the collected SIP attack traffic. Sensors using these signatures can be applied for real-time attack detection and mitigation.

REFERENCES

[1] (2013, Jan) Rosenberg, et. al., “rfc3261.pdf” June 2002. [Online]. http://tools.ietf.org/pdf/rfc3261.pdf.

[2] Dirk Hoffstadt, Alexander Marold, and Erwin Rathgeb, “Analysis of SIP-Based Threats Using a VoIP Honeynet System,” in *Conference proceeding of the 11th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom-2012)*, Liverpool, UK, 2012.

[3] Dirk Hoffstadt, Stefan Monhof, and Erwin Rathgeb, “SIP Trace Recorder: Monitor and Analysis Tool for Threats in SIP-based Networks,” *Wireless Communications and Mobile Computing Conference (IWCMC), 2012 8th International*, August, 2012.

[4] (2013, Jan) Sipvicious. [Online]. http://blog.sipvicious.org

[5] (2012, Nov) Wireshark. [Online]. http://www.wireshark.org

[6] (2012, Nov) Digium, Inc., “Asteric IP PBX, VoIP Gateway, IVR & Open Source Communications”. [Online]. http://www.asterisk.org/

[7] Sven Ehlert, “Denial-of-Service Detection and Mitigation for SIP Communication Networks,” PhD Thesis, October 2009.

[8] Iyatit Mokube, Michele Adams, “Honeypots: Concepts, Approaches, and Challenges,” *ACM-SE 45 Proceedings of the 45th annual southeast regional conference, 2007.*

[9] Mohamed Nassar, Saverio Niccolini, Radu State, and Thilo Ewald, “Holistic VoIP Intrusion Detection and Prevention System,” *Proceedings of the 1st International conference on Principles, systems and applications of IP telecommunication (IPT Comm,)*, 2007.

[10] Mohamed Nassar, radu State, and Olivier Festo, “VoIP Honeypot Architecture,” *in 10th IFIP/IEEE International Symposium on Integrated Network Management*, Munich, Germany, 2007.

[11] Craig Valli, “An Analysis of Malfeasant Activity Directed at a VoIP Honeypot,” *Proceedings of the 8th Australian Digital Forensics Conference*, 2010.

[12] (2013, Jan) SQLite. [Online]. http://www.sqlite.org/

[13] (2013, Jan) JAIN. [Online]. http://jsip.java.net/