Improved Detection and Correlation of Multi-Stage VoIP Attack Patterns by using a Dynamic Honeynet System

Dirk Hoffstadt, Niels Wolff, Stefan Monhof, Erwin Rathgeb
Computer Networking Technology Group
University of Duisburg-Essen
Essen, Germany
dirk.hoffstadt@iem.uni-due.de, niels.wolff@stud.uni-due.de, stefan.monhof@uni-due.de, erwin.rathgeb@iem.uni-due.de

Abstract - **Security issues like service misuse and fraud are well-known problems of SIP-based networks. To develop effective countermeasures, it is important to know how these attacks are launched in reality. For gathering the required data, a specialized SIP Honeynet System has been running since January 2009 and has recorded over 58 million SIP messages. The analyses have shown that SIP-based misuse is typically performed as a multi-stage attack and the IP address of the attacker changes before the actual Toll Fraud calls. To be able to correlate all attack stages despite intermediate changes of the attacker's IP address we developed the new Dynamic Honeynet System (DHS), which reacts according to the attackers' behaviour and uses a dynamic Honeypot configuration in real-time to significantly improve the detection efficiency. We present the architecture and new features such as dynamic reconfiguration and demonstrate its attack correlation capabilities. We developed a Sensor component to realize this system. The Sensor provides active monitoring based on signatures to detect attacks in real-time and controls the dynamic Honeypot.**

Keywords: SIP; honeynet; misuse; fraud; attacks; VoIP; security

I. Introduction

Voice-over-IP (VoIP) communication based on the Session Initiation Protocol (SIP) [1] has evolved as 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. SIP servers, especially if they allow access from external networks, are subject to 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 benefit by making toll calls (international, cellular, premium services) via third party accounts. This attack is called Toll Fraud and can cause the account owner substantial financial damage in a very short time (e.g., 11 million euro in five months [2]). In [3] we presented our VoIP Honeypot analysis results and showed that attackers change their IP addresses during an attack. Therefore, we could not identify one attacker across all attack stages (see Section II). To solve this issue, we developed the Dynamic Honeynet System (DHS) with an active monitoring component (so-called Sensor) to detect attacks in real-time (see Section IV). If an attack is detected, the DHS is controlled by the Sensor component to modify the configuration and to save the attacker's credentials for identification at a later date. In addition, the system allows to set up new Honeypots, i.e. SIP servers and VoIP accounts, in real-time in reaction to an attacker's scanning behaviour. This function significantly improves the number of observable attacks.

II. VoIP-Specific Misuse

SIP is used to establish sessions (e.g., voice, video) between two user agents. For the purpose of this paper, the following message types are relevant: If a user agent (i.e., SIP device) wants to establish calls via a provider voice server in SIP-based networks, a registration at an infrastructure component 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 UNAUTHORIZED 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. 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. As the standard specifies that an OPTIONS packet must be answered, regardless of its source or existing connections we have to ensure that this communication is always possible. Furthermore, an INVITE message is sent by a user agent to initiate a session.

Typically four distinct attack stages are necessary to exploit a third party SIP extension. This attack behaviour we have also observed during our previous VoIP Honeynet study [3]. The first three stages are performed to identify and hijack regular SIP extensions and can be executed by using freely available tool suites. A common white-hat attacking tool for SIP is the open source tool suite Sipvicious [4]. During the first three attack stages the attacker's source IP address does typically not change. Therefore, we can identify the attacker by using the source IP address. The fourth attack stage, the actual Toll Fraud Call, occurs after an arbitrary time period and the attacker uses different source IP addresses. Thus, the attacker cannot be identified by its source IP address anymore.

Figure 1 shows a typical example attack recorded on March, 24th, 2011, which involved a total of 58,216 SIP messages and resulted in 24 Toll Fraud attempts. As first attack stage, a Server Scan was performed. The attacker pinged IP addresses with OPTIONS packets to identify SIP devices. Then, to identify extensions the detected SIP server, the attacker tried to register at several extensions without password. This is called Extension Scan. An extension identifier consists of digit sequences and/or series of letters. If the extension exists, the server normally answers with a "401 UNAUTHORIZED", because the attacker does not initially use a password. If it does not exist, a "404 NOT FOUND" is typically returned. The result of this attack stage is a complete list of all existing extensions (provider accounts). In the third stage, the attacker has to guess the password of identified extensions with a Registration Hijacking attack. The attacker sent numerous REGISTER messages to the SIP server with different passwords to register at a given extension. If the password is guessed, the information is stored to register at this extension later on.

As shown in Figure 1, the Toll Fraud Call was launched after a significant period of time and the source IP address had changed. In terms of SIP messages, the attacker first sends a REGISTER message with the correct password. After receiving the "200 OK" message from the server, the attacker can initiate calls by using INVITE messages. As it is not feasible to identify the attacker by its source IP address we have to use our new identification approach (see Section V) to correlate all these events despite the time lag and address change.

III. Related Work

A detailed analysis of VoIP attacks against Honeypots is given by Valli [5]. The source data is captured by a Honeypot system consisting of several virtualized Low Interaction Honeypots that are logging to the same system. A simple statistical analysis is performed. The use of Sipvicious as tool is proven and another tool, called "sipsscuser", is found. The author speculates that the behaviour of sipsscuser points to a botnet- or worm-like activity. The above described Honeypots and analyses only capture data at the specific Honeypots. But with the knowledge of the workflow of tools like Sipvicious, a broader view is required: It is necessary to analyze a complete network with or without SIP devices to identify all attack patterns. Due to the fact that there are only analyses with a few Honeypots, it is important to determine if the attack attempts are local problems (e.g., one host) or if whole subnets are attacked.

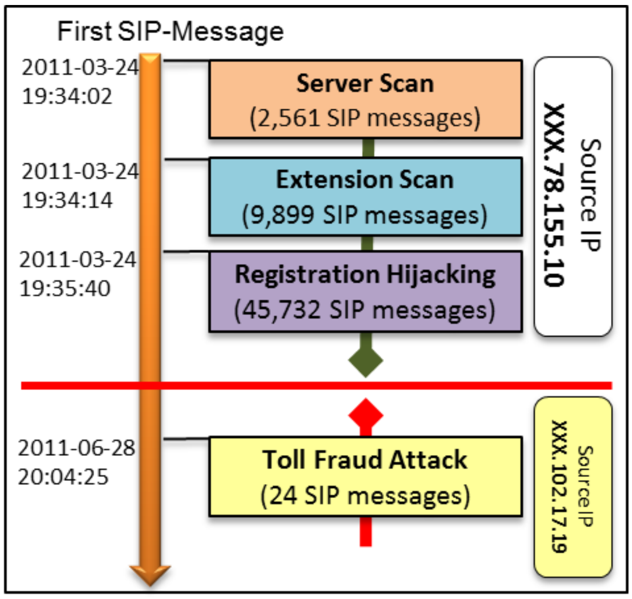
First SIP-Message
2011-03-24 19:34:02
Server Scan (2,561 SIP messages)
2011-03-24 19:34:14
Extension Scan (9,899 SIP messages)
2011-03-24 19:35:40
Registration Hijacking (45,732 SIP messages)
Source IP XXX.78.155.10
2011-06-28 20:04:25
Toll Fraud Attack (24 SIP messages)
Source IP XXX.102.17.19

Figure 1: Multi-stage Attack

The Dynamic Honeynet System presented in this paper is based on the Dionaea-Honeypot-Framework [6] which can be used as Low or Medium Interaction Honeypot [7]. Dionaea provides different simulated services such as SMB, HTTP, FTP or SIP. Our test setup requires the SIP function only. As the Dionaea SIP component does not provide all SIP methods (01/2011) like REGISTER, this implementation is not sufficient to detect e.g., a Sipvicious attack. Due to the fact that there are unsupported SIP functions and performance issues, we developed a new SIP component which is completely integrated in the Dionaea Honeypot framework and has low hardware requirements. The new implementation supports all necessary interfaces and simulates all SIP methods. Therefore, this component reacts as a standard SIP server and is an interesting-looking SIP device for attackers.

In [3] we present our initial SIP-based Honeypot System. We show that it is not feasible to identify an attacker over the whole attack chain (attack stages one to four) because attackers typically change their IP addresses before starting a stage four attack. This issue is solved by the Dynamic Honeynet System, which is presented in Section IV. In [8] we present our SIP Trace Recorder (STR), which allows passive attack monitoring in SIP-based networks. Due to the fact that we have to configure our honeypots in real-time, it is necessary to detect an attack and to send a notification message to the attacked honeypot. To solve this issue, we developed the Sensor component (see IV), which does active monitoring based on signatures and sends alarm messages.

Furthermore, in [3] we use a static Honeypot configuration with predefined SIP accounts. The new implementation provides dynamic functions based on the attacker's behaviour and controlled by the Sensor component which we call Enable Extension Function (EEF).

IV. Dynamic Honeynet System Architecture

The new Dynamic Honeynet System consists of two parts: The active monitoring Sensor and the Dionaea-based Low Interaction Honeypot component. Furthermore, Figure 2 shows that a secure interface between Sensor and Honeypot is used to realize the Honeypot reconfiguration if an attack is detected.

A. Sensor

1) Concept

The Sensor component is a software tool for rule-based detection and reporting of misuse in SIP-based VoIP networks. It recognizes sequences of SIP messages that are characterized by rules and can report information about recognized message sequences, their source and their destination to another system via a defined interface. This interface is used for

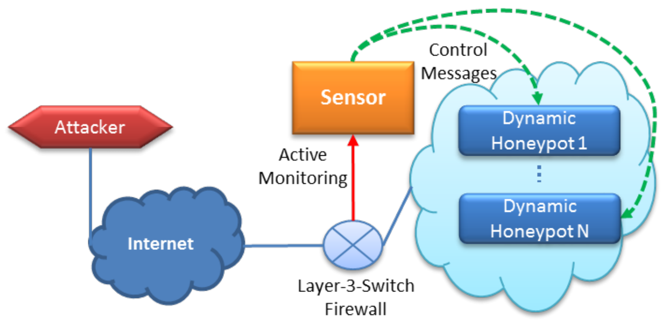

Figure 2: Dynamic Honeynet System

communication between the Low Interaction Honeypot and the Sensor. We implemented the Sensor component in C++ using libpcap [9] for easy access to network interfaces, filtering of network traffic and platform independency. Libpcap can be used on different platforms and on devices with limited hardware resources like home gateways.

The process of misuse detection and reporting can be separated into three different phases: First capturing and filtering of SIP messages, second analysis of the captured data and third reporting.

In this setup only one Sensor component is used. It is attached to the monitor port of the layer 3 switch that is connected to our honeypots (see Figure 2). Therefore, it receives all data that is sent to any of the Honeypots. A pcap filter is used to filter SIP messages out. In the analysis phase timing conditions between messages are also considered for the recognition of SIP message sequences. Each rule defines a specific pattern of SIP messages and timing conditions. A SIP message definition in a rule can contain information about destination and source of the message and SIP header values. Comparison of these parameters with a fixed value, among different messages of the rule or within one message is possible. When a rule matches, an action is performed. In this setup the Low Interaction Honeypot is contacted to enable the attacked extension(s) for the attacker.

2) *Architecture*

The three phases described above are represented by three different modules of the Sensor component. As shown in Figure 3, the first module is called “Listener”. It retrieves all SIP messages from a network interface in promiscuous mode and enqueues them into a FIFO-queue. The SIP messages are dequeued by the “Analyzer” module that is also aware of the message sequences of well-known attacks (rules). One rule is defined for a sequence of messages. The Analyzer compares every received SIP message to the first message of every rule. If a received message matches this first message, the rule is copied and updated so that messages received later are compared with the next message of the rule. One message can lead to the update of more than one rule state. If the inspection of the messages fails, the rule is not updated. If a message matches no message of a rule, it is discarded, else it is stored for comparison with messages received later. Since every rule has at least one time condition that specifies in which period all messages of the sequence have to be received, the Analyzer is able to delete irrelevant messages and rule states. So the memory usage of the Analyzer can be minimized (though it depends on the defined rules and the amount of received messages). A rule matches if a received message was successfully compared to the last message of the rule. Then, the third module “Notification” triggers the Honeypot using an interface that is described in section C.

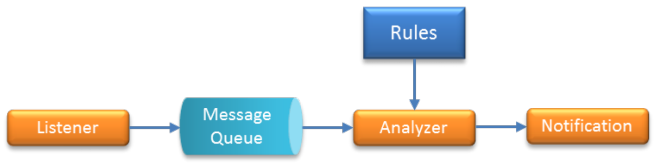

Figure 3: Architecture of the Sensor Ccomponent

B. *Low Interaction Honeypot*

The Low Interaction Honeypot is based on the Dionaea Framework [6], which is implemented in Python [10]. This script-based component provides SIP service only (TCP and UDP), but with comprehensive processing and basic functionality of all SIP requests (INVITE, REGISTER, OPTIONS, ACK, BYE, CANCEL) according to the SIP RFC 3261 [1]. This component is used to simulate a SIP server. Moreover, we compared the behaviour of the widely used open source VoIP server Asterisk [11] with this script-based component and optimized its operations so that it resembles Asterisk more closely.

We are able to configure up to 10,000 extensions within a freely selectable number range or with any username string. The Honeypot provides three different operations modes:

- A complex password can be configured and the Honeypot allows successful registrations if the password given is correct, e.g., after a successful registration hijacking attempt. This mode emulates the “normal” behaviour of a server.
- No password is configured and the Honeypot denies all registration attempts. This mode is used to fully observe Registration Hijacking attempts until the attacker gives up.
- Based on an activation trigger from the Sensor component, the Honeypot accepts the next registration attempt and saves the password used (IV.C). This mode is used to efficiently capture Registration Hijacking attempts after a relatively small number of SIP messages (e.g., 100) and to provoke the maximum number of Toll Fraud attempts (because no Hijacking attempt fails).

The Honeypot component can be configured to emulate different user agents (e.g., Asterisk [11], SER [12]) and their specific behaviour patterns. With regard to authentication, the SIP-Digest method [1] is implemented with fixed or random-generated, complex passwords. The Honeypot extension accepts Toll Fraud calls if its configured password is cracked by a Registration Hijacking attack (stage 3) or the Sensor component sends an activation message (see Section V.B) due to a detected attack. A local database stores the current configuration, any modification made by the Sensor component and the account credential values, which belong to an attack.

Thus, a system restart or configuration transfers to another Honeypot are possible without losing information.

If an attacker has successfully registered at an extension, the Honeypot accepts Toll Fraud calls and uses an echo channel for audio communication. The channel is dropped after a random period of time (10 to 60 seconds). Moreover, other behaviours are feasible such as voicemail or call transfer to an IP telephone (real user agent).

The hardware requirements are rather low compared to a productive PBX. Our test virtual machine has one CPU core, 384MB RAM, and a 2GB hard disc and supports over 520,000 SIP requests per hour.

C. Interface between Sensor and Low Interaction Honeypot

The Sensor and the Low Interaction Honeypot component are connected to a local private virtual control network for exchanging control messages. This network is used exclusively by these components and only for the exchange of control messages. Furthermore, it is not reachable from the Internet.

For use in combination with the Low Interaction Honeypot component we defined different Sensor rules to control the Honeypot behaviour according to the attack stages. In particular, regarding to a stage three attack we configured a rule that matches if 100 REGISTER requests from one source IP, to one of the honeypot IPs, containing only one extension (to header) are received in a time span of 60 seconds. This signature is based on a comprehensive evaluation of the attack data collected in our Honeynet over last three years and provides a reliable detection of Registration Hijacking attempts. The parameters can be adapted if the on-going monitoring of attacks should indicate changes of the attacker's behaviour. After the rule matches the attacker's IP and the attacked extension ID are sent to the Honeypot component in order to enable the extension for this attacker.

The notification is sent in control messages via the local virtual network. These control messages are SIP NOTIFY requests with a custom XML structure in the message body. Figure 4 shows an example of a control message. The Sensor component at 192.168.99.94 has detected a Registration Hijacking attempt (method="auth") of the extension 5994 from the source IP 132.252.151.50 to the honeypot at 192.168.99.105 (external address is not contained in the control message). If an attack is detected the Sensor component sends a SIP NOTIFY message to the attacked Honeypot host.

```
NOTIFY sip:honeypotreport@192.168.99.105 SIP/2.0
To: <sip:honeypotreport@192.168.99.105>
From: <sip:sensor@192.168.99.94>
Call-ID: 3027823560
CSeq: 282 NOTIFY
Content-Type: text/plain
<?xml version="1.0"?>
<dialog-info proto=SIP>
<state>method="auth"</state>
<param host="132.252.151.50" username="5994"></param>
</dialog-info>
```

Figure 4: Control Message

A Dionaea script that checks the source, message type, XML structure and commands of every received SIP message is listening to the network interface of the local virtual network at the Low Interaction Honeypot component. If a message is valid, it is forwarded to the default SIP Honeypot script via the loopback interface. The message is recognized as an internal control message and the commands of the XML body are executed so that the specified extension is enabled for the attackers IP address. If an extension is enabled for one attacker, it will not be enabled for any other IP address of an attacker again. Details are described in section V.B.

V. NEW FEATURES

A. Dynamic Configuration

The default configuration of the Low Interaction Honeypot component can be used to set up the IP addresses and extensions that will be provided by the Honeypots at start-up. All further behaviour parameters are controlled by the Sensor component. If a Server Scan is detected in one network, the Sensor activates more IP addresses by using the interface to the Honeypot components. If an attacker performs an Extension Scan, we are able to dynamically provide more extensions on the target Honeypot. This optimization enhances the detection capabilities and allows efficient adaption of the monitoring scope to capture the maximum number of attacks with minimum resource use. This is necessary because an attack only continues if an attacker discovers active SIP devices [3].

In contrast to previous Honeypots with static configuration, we only use extensions that have to be enabled by the Sensor component in our current setup, i.e. there are no enabled extensions with predefined passwords. In this constellation it is not possible to successfully authenticate with an extension or perform any registered action (e.g. INVITE) without the decision of the Sensor component to enable an extension for a stage three attack. The following function is essential to correlate different attacker identities between attack stages three and four.

B. Enable Extension Details

We developed the "Enable Extension Function" (EEF) due to the fact that for further analysis a successful authentication is necessary. If the Sensor component detects a Registration Hijacking attack, the EEF activates the authentication option for the specific extension based on a threshold value:

1. For SIP authentication a challenge-based mechanism based on HTTP Digest authentication is used [1]. When the attacker tries to crack an identified extension, his user agent calculates a response hash based on a number of data (nonce value, username, SIP method, SIP URI) including a secret (the extension password). The nonce value is a random base64 value generated by the Honeypot.

2. In case of a stage three attack from one attacker to a single Honeypot extension (identified by identical source IP address, destination IP address, destination extension and username) with an amount of registrations attempts exceeding the threshold value (e.g., 100), the EEF will setup the extension for successful registration with the password probed in the next register attempt. All

authentication values are stored in a local database according to enabled extensions. The mentioned threshold value can be adjusted to masquerade our Honeypot. Thus, we try to avoid that the attackers recognize the Honeypot behaviour.

3. At this time, the generated nonce value will be used for every registration response concerning this specific extension.
4. After activating the authentication for an extension, the Honeypot replies with a successful response (200 OK) to the attacker if the attacker's response hash is equal to the one stored in the Honeypot database. Only one authentication option will be saved in the database for every stage three attack. It is not possible to activate more than one password for a specific extension. Therefore, we are able to identify an attacker without using the source IP address.
5. It is feasible to successfully establish a call with an authenticated INVITE request in correlation with an already enabled extension (known response hash).
6. A successful registration attempt at an enabled extension is possible for up to ten REGISTER messages, only. After these attempts, the system prevents further authentication attempts with an error message because probing of response hashes must not possible. Therefore, only one attacker is mapped to an extension that uses the same authentication response.

With the Enable Extension Function it is possible to reliably identify attackers between different attack stages.

VI. Evaluation

From August, 1st till September, 9th, 2012 our Dynamic Honeypot was connected to the Internet and analysed the attack behaviour. The Honeypot received 1,283,523 SIP messages that belonged to 23 different source IP addresses. Per single attack the attackers performed Server and Extension Scans with up to 12,000 messages. Server Scans did not stop if an active server was identified. In the first setup our dynamic Honeypot created a new extension for any SIP request which belongs to an Extension Scan. However, this resulted in a lot of extensions (up to 10,000) and in massive Registration Hijacking attempts with up to 504,069 packets per attacker. Therefore, we defined a threshold for creating extensions (> 10) and a configuration with 100 extensions that are randomly generated at start-up. If an extension was detected, the attacker tried to guess the password. Due to the EEF threshold the attacker successfully registered an existing extension after 100 SIP messages.

After a significant period of time and a successful registration at a hijacked extension, Toll Fraud calls with a small number of SIP messages (up to 48) were established at various time points. Due to the fact that Registration Hijacking attacks occurred, extensions were activated by the EEF for successful registration. The following describes one example attack in detail and is shown in Figure 5.

On August, 23th, 2012, at 13:01:45 our monitoring recognized a Server and Extension Scan (see Figure 5, A) with

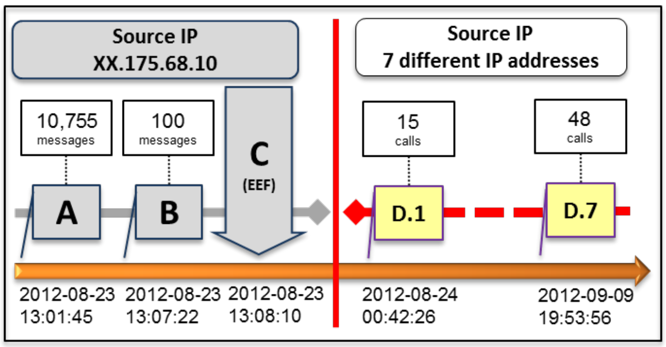

Figure 5: Dynamic Extension Enabling and Toll Fraud Calls

10,755 messages. Afterwards, at 13:07:22 a Registration Hijacking attack was performed (B) and the Sensor component sent a notification to the Honeypot due to the signature threshold of 100 SIP REGISTER messages (C). The EEF activated the attacked extension 9903 as described in Section V.B at 13:08:10. Therefore, the attacker could successfully register the detected extension. However, there was no Toll Fraud attempt at that time. The source IP address did not change until the successful registration.

On August, 24th, 2012, at 00:42:26, the hijacked extension 9903 was directly registered by an attacker by using the "correct" credentials. Only five seconds later, the first of seven Toll Fraud calls was performed (see Figure 5, D.1). It is important to notice, that the attacker used another source IP address. At this time the Honeypot detected 15 call attempts to the same target telephone number, but the attacker probed different dialling prefixes. Moreover, further Toll Fraud calls occurred and the attacker tried to establish up to 48 calls to different telephone numbers by using different source IP addresses from August, 29th until September 9th (D.7). This example clearly demonstrates the benefits of the new Dynamic Honeynet System, because the correlation between Registration Hijacking attempts and subsequent Toll Fraud attempts could not be proven with the previously known setups.

VII. Conclusion

In this paper, we have presented the Dynamic Honeynet System that is used to identify an attacker during all attack stages independently of the source IP address. This functionality is necessary because previous analysis results show that attackers typically change their IP addresses during an attack. To realize this system, we developed the new Sensor component, which provides active monitoring based on signatures to detect attacks in real-time and controls the dynamic Honeypot. The lightweight Honeypot component provides multiple attack targets with low hardware resources and dynamic, real-time configuration to setup additional SIP servers and extensions. The system offers efficient adaption of the monitoring scope to capture the maximum number of attacks with minimum resource use. Moreover, the EEF provokes maximum number of Toll Fraud attempts by avoiding unsuccessful Registration Hijacking and reduces the message overhead (high volume Registration Hijacking).

Finally, the EEF allows the correlation of stage 4 attacks based on authentication, but independently of timing or IP addresses.

A first field test in our network environment demonstrates the benefit of our new approach which traces the attacker's behaviour regardless of the source IP address. If an extension is hijacked, we detected several successful registrations and call attempts to different telephone numbers established from different source IP addresses. Due to this behaviour, there could be a shared attacker memory. To substantiate our results, the Dynamic Honeynet System is still running and will collect further attack data. We will test different thresholds for the dynamic behaviour functions. Moreover, we intend to release the Dynamic Honeynet System under an open source license.

We are currently developing a distributed SIP misuse detection system based on the signatures generated during our Honeypot study. This system will be deployed at different locations which allow distributed attack detection and a comparison of different networks in the internet. This work is part of a German research project called Sunshine [13] to develop VoIP detection and mitigations components.

REFERENCES

[1] J. Rosenberg et al., RFC 3261-SIP: Session initiation protocol, 2002.

[2] (2010, Dec.) Sipvicious Blog. [Online]. http://blog.sipvicious.org/2010/12/11-million-euro-loss-in-voip-fraud-and.html

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

[4] (2011, Aug.) Sipvicious. [Online]. http://blog.sipvicious.org

[5] Craig Valli, "An Analysis of Malfeasant Activity Directed at a VoIP Honeypot," in *Proceedings of the 8th Australian Digital Forensics Conference*, Perth, Australia, 2010, pp. 168-174.

[6] (2012, Aug.) dionaea. [Online]. http://dionaea.carnivore.it/

[7] Iyatiti Mokube and Michele Adams, "Honeypots: concepts, approaches, and challenges," in *Proceedings of the 45th annual southeast regional conference*, Winston-Salem, North Carolina, USA, 2007, pp. 321-326.

[8] Dirk Hoffstadt, Stefan Monhof, and Erwin P. Rathgeb, "SIP Trace Recorder: Monitor and Analysis Tool for threats in SIP-based networks," in *TRaffic Analysis and Classification Workshop (IWCMC2012-TRAC)*, Limassol, Cyprus, Aug. 2012.

[9] WinPcap. [Online]. http://www.winpcap.org

[10] Python Software Foundation. (2012, Aug.) Python Programming Language. [Online]. http://www.python.org/

[11] (2011, Aug.) Asterisk. [Online]. http://www.asterisk.org

[12] iptel.org. (2012, Aug.) SIP Express Router. [Online]. http://www.iptel.org/ser/

[13] (2012, Sep.) BMBF Sunshine Project. [Online]. http://www.sunshineproject.net