PhD Dissertation

International Doctorate School in Information and Communication Technologies

Department of
Information Engineering and Computer Science
University of Trento, Italy

Analysis and Protection of SIP based Services

Raihana Ferdous

Advisor:

Prof. Renato Lo Cigno

University of Trento, Italy

May 2014

To my father and mother

Abstract

Multimedia communications over IP are booming as they offer higher flexibility and more features than traditional voice and video services. IP telephony known as Voice over IP (VoIP) is one of the commercially most important emerging trends in multimedia communications over IP. Due to the flexibility and descriptive power, the Session Initiation Protocol (SIP) is becoming the root of many sessions-based applications such as VoIP and media streaming that are used by a growing number of users and organizations. The increase of the availability and use of such applications calls for careful attention to the possibility of transferring malformed, incorrect, or malicious SIP messages as they can cause problems ranging from relatively innocuous disturbances to full blown attacks and frauds.

Given this scenario, a deep knowledge of the normal behavior of the network and users is essential to problem diagnosis and security protection of IP Telephony. Moreover, analysis tools taking into account service semantics and troubleshooting VoIP systems based on SIP are of paramount importance for network administrators. However, efficient design and deployment of robust and high performance security controlling systems remain a high challenge, in particular due to the open architecture of the Internet, heterogeneous environment and real time communication constraint.

This thesis deals with the analysis and protection of services based on the SIP protocol with a special focus on SIP based VoIP applications. The first part of the work is dedicated to the conformance and security analysis *of SIP based VoIP services. To this end, our first endeavor is to define a formal conceptual model of VoIP threat domain with the aim to exchange a common vocabulary about the security related information of the domain. We have introduced an* ontology *defined as "VoIP-Onto" that provides a formal representation of a comprehensive taxonomy of VoIP attacks followed by specific security recommendations and guidelines for protecting the underlying infrastructure from these attacks. The use of "VoIP-Onto" is not only limited to as a general vocabulary and extensible dictionary for sharing domain knowledge about VoIP security, but also can be employed in a real environment for testing or intrusion detection purposes.*

We have also concentrated on designing synthetic traffic generators considering the difficulties and challenges of collecting real-world VoIP traffic for the purpose of testing monitoring and security controlling tools. To this end, we have introduced "VoIPTG", a generic synthetic traffic generator, that provides flexibility and efficiency in generation of large amount of synthetic VoIP traffic by imitating the realistic behavior profiles for users and attackers. We have also implemented "SIP-Msg-Gen", a SIP fuzzer, capable to generate both the well-formed and fuzzed SIP messages with ease.

Then, we focus on designing an on-line filter able to examine the stream of incoming SIP messages and classifies them as "good" or "bad" depending on whether their structure and content are deemed acceptable or not. Because of the different structure, contents and timing of the SIP "bad" messages, their filtering is best carried out by a multistage classifier consisting of deterministic lexical analyzer and supervised machine learning classifiers. The performance and efficiency of our proposed multi-stage filtering system is tested with a large set of SIP based VoIP traffic including both the real and synthetic traces. The experimental result of the filtering system is very promising with high accuracy providing fast attack detection.

Next, the focus is shifted on the understanding and modeling the social *interaction patterns of users of the VoIP domain. The notion of "social networks" is applied in the context of SIP based VoIP network, where "social networks" of VoIP users are built based on their telephone records. Then, Social Network Analysis (SNA) techniques are applied on these "social networks" of VoIP users to explore their social behavioral patterns. A prototype of filtering system for SIP based VoIP services is also implemented to demonstrate that the knowledge about the social behavior of the VoIP users is helpful in problem diagnosis, intruders detection, and security protection. The filtering system is trained with the normal behavioral patterns of the users. The machine, thus trained, is capable of identifying "malicious" users.*

Keywords: VoIP, Session Initiation Protocol (SIP), Traffic analysis, Anomaly Detection, Social Behavior Analysis, Supervised Learning, SVM, Clustering

Acknowledgments

The accomplishment of this work would not have been possible without the collaboration and support of many kind people around me, and this page is here to thank them all.

I would like to start by remembering Prof. Yoram Ofek. May he rest in peace.

First, I would like to express my sincere gratitude to my advisor, Prof. Renato Lo Cigno, for being an excellent mentor for so many years. His vision, knowledge, motivation, patience and support have been a source of inspiration and have always pushed me ahead in the path of personal and professional growth and maturation. I am deeply grateful to him for giving me the moral support and the freedom to concentrate on my research. I would also like to thank him for taking quick initiatives to solve various problems that I encountered during my stay in Trento.

I would like to express my deep appreciation to Prof. Alessandro Zorat for his insightful guidance and support in my research. Your advice, constructive comments and warm encouragement are invaluable. It has been an honor and pleasure to work with you.

I am deeply grateful to Dr. Andrea Mongera for allowing us to collect SIP traces without which this research work would not have been possible. Special thanks also to him for the wonderful opportunity to work in "Sistemi Informativi, Servizi e Tecnologie Informatiche" and for providing funding during the last year of my research. I would also like to express my gratitude to Dr. Alessandro Villani for the guidance and help during my work period in SISTI.

I would like to thank Marco Nesler, Andrea Avi, Angelo Giacomozzi and Massimiliano Fiorazzo for their support in collecting SIP traces and also during my work period in SISTI.

A special thanks goes to Roberta, Csaba, Alessandro, and Sasha for their support in these years.

I would like to thank my parents for their love, care and encouragement. All the support they have provided me over the years is the greatest gift of my life. And most of all, I would like to thank my husband Shameem for his love, support, and sacrifices, and for keeping me motivated throughout the completion of my PhD.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Multimedia Communications Over IP

Multimedia communications over Internet Protocol network have become increasingly important due to the simplicity and flexibility of packet switching using the IP protocol. IP telephony, also known as Voice over IP (VoIP), is one of the commercially most important trends in multimedia communications over IP. It enables people to use the Internet as the transmission medium for voice communications by sending voice as data packets over private or public IP networks as well as reassembling and decoding at the receiving end. VoIP services are generally based on standard protocols for signaling (i.e., Session Initiation Protocol (SIP), H.323, and Media Gateway Control Protocol (MGCP)) and transporting media, albeit solutions like Skinny[1] by Cisco and the Open Source IAX[2] adopted by Asterisk project are very popular in enterprise solutions. A diagram of a typical VoIP network is presented in Fig. 1.1.

SIP seems to be the most promising candidate as the signaling protocol for future IP-based telephony services. It has also been chosen by the

[1]Skinny Call Control Protocol,
`http://www.cisco.com/c/en/us/tech/voice/skinny-call-control-protocol-sccp/index.html`

[2]IAX: Inter-Asterisk eXchange, `http://tools.ietf.org/html/rfc5456`

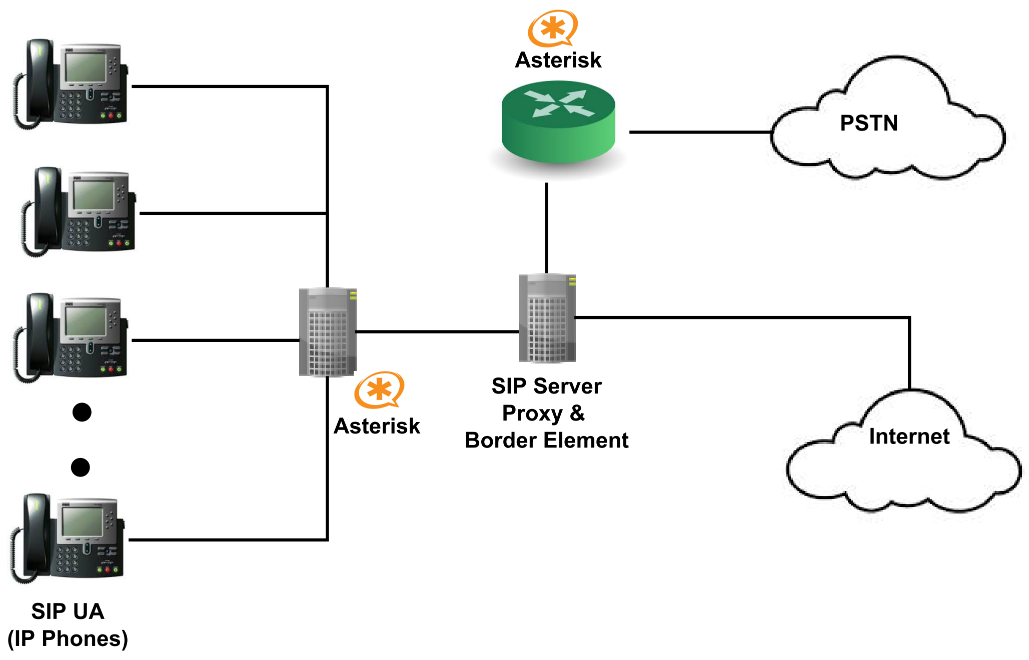


Figure 1.1: A typical campus VoIP network

Third-Generation Partnership Project (3GPP) as the protocol for multimedia application in 3G mobile networks [61]. Due to its flexibility and descriptive power, SIP is becoming the support not only for Voice over IP and Internet telephony, but also for many other so called session-based applications, such as most multimedia streaming, chats, and many others integrating web-services with telephone and voice. Particularly, SIP based VoIP opens up exciting possibilities for both businesses and individuals by lowering the cost and increasing the flexibility in communication and operation.

1.2 Session Initiation Protocol

SIP (RFC 3261 [156] and related documents) is an application-layer signaling protocol for creating, modifying, and terminating multimedia sessions between two or more participants (could be a simple two-way telephone call or a collaborative multimedia conference session) in an IP based network. It is a request-response based protocol.

1.2.1 SIP Components

The SIP protocol follows the client/server model whose basic components are:

- **User Agents (UA)**: A SIP user agent is a logical network end-point used to create or receive SIP messages and thereby manage a SIP session. User agent itself has a client element, the *User Agent Client (UAC)* and a server element, the *User Agent Server (UAS)*. The *UAC* is responsible to create requests, and the *UAS* processes and responds to each request generated by a *UAC*. SIP UA can be lightweight clients suitable for embedding in end-user devices such as mobile handsets or PDAs, alternatively, they can be desktop applications (e.g., soft-phones).

- **SIP Servers**: The main function of the SIP servers is to help user agents to establish sessions or to perform other functions by providing name resolution and user location, and to pass on messages to other servers using next hop routing protocols.

  - **Proxy Server**
    The proxy server is an intermediary entity that acts as both a server and a client for the purpose of making requests on behalf of other clients. A proxy server primarily plays the role of routing, meaning that its job is to ensure that a request is sent to another entity closer to the targeted user agent. When a user agent client wants to establish a call, it does not know where

callee's user agent server is located. It only knows where is the out-bound proxy of the callee, so it relays the request to it. The outbound proxy can directly deliver the request if it knows recipient's address otherwise, through Domain Name System (DNS) lookup over the SIP URIs, it will discover the incoming proxy for the callee and forwards the request.

– **Location Service**
  A location service is used by a SIP redirect or proxy server to obtain information about a callee's possible location. For this purpose, the location service maintains a database of SIP-address/IP-address mappings.

– **Redirect Server**
  The redirect server is used during session initiation to determine the address of the called device and it informs the caller's UA about the next hop server. The caller's UA then contacts the next hop server directly.

– **Registrar Servers**
  User Agents contact registrar servers to announce their presence in the network. The SIP registrar server is a database containing locations and users preferences as indicated by the *User Agents*. A registrar server accepts SIP registration requests and binds the information it receives (the SIP address and associated IP address of the registering device).

1.2.2 SIP Addresses

Users in a SIP environment are identified by SIP Uniform Resource Identifiers (URIs). The format of a SIP URI is similar to an e-mail address, generally consisting of a username and a domain name. In turn, a SIP URI is mapped with a terminal address. The username part can be a string or a number, for instance:

sip:bob@chicago.com
sip:1234567890@chicago.com

1.2.3 SIP Message Format

SIP is a text-based protocol, the information exchanged within the protocol is encoded as strings of characters. SIP messages are divided into lines of characters where a line is a series of characters that is delimited with the two characters *Carriage Return* and *Line Feed* (CRLF).

There are two types of SIP messages: requests and responses. Both of them consist of a start line, one or more header fields, an empty line indicating the end of the header fields, and an optional message body. Header fields are composed of a field name, followed by a colon (":"), followed by the field value, and terminated by CRLF. The body of a message is simply lines of characters.

A SIP message contains two types of headers: (i)mandatory, (ii)optional. A specific header field can be mandatory for a particular type of message while it is optional for others. For example, the *Contact* header field is mandatory in an INVITE request message while it is optional for other types of request messages. Also, some headers can only appear once in a message (e.g., *From, To, Call-ID, etc*); multiple occurrences of them in the same message should be flagged as an error and the corresponding message should be discarded. Unfortunately, some of these rules are not fully obeyed in all implementations of the SIP protocol, and therefore it is possible for "erroneous" messages to be transferred from node to node over the network.

Syntax of the SIP messages are defined by a context-free grammar speci-

Table 1.1: SIP Request Methods

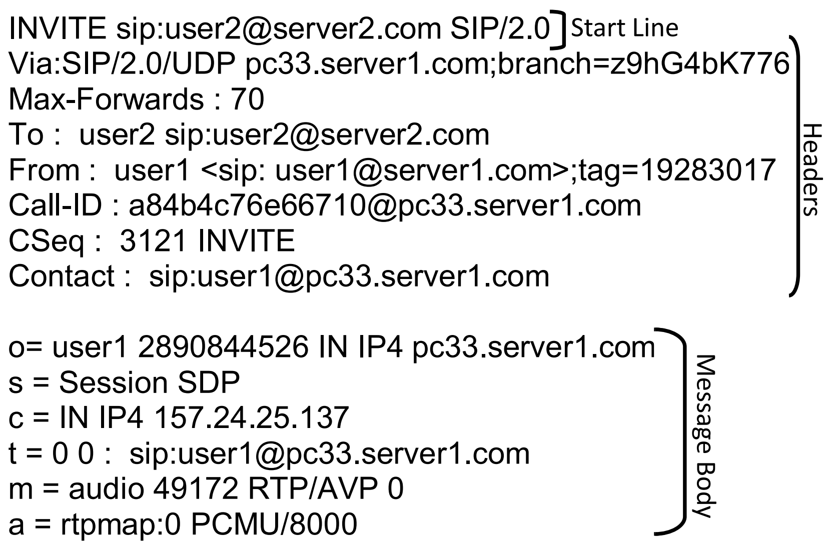
Request Method	Description
INVITE	To initiate a call to the other party
ACK	To respond to a corresponding INVITE request
BYE	To terminate a call
OPTIONS	To obtain information about the capabilities of servers
REGISTER	To register a user account
CANCEL	To abort any pending request
PRACK	To sent provisional acknowledgment
SUBSCRIBE	To subscribe for notification from the notifier
NOTIFY	To notify the subscriber of a new event
PUBLISH	To publish an event to the Server
INFO	To send mid-session information
REFER	To ask the recipient to issue call transfer
MESSAGE	To transport instant messages
UPDATE	To modify the state of a session

fied in Augmented Backus-Naur Form (ABNF) [35], a metalanguage based on Backus-Naur Form (BNF). As an example, the general structure of a SIP message is defined as:

```
generic-message = start-line *message-header | CRLF [message-body]
start-line      = request-line | status-line
```

This indicates that a SIP generic-message consists of a `start-line`, followed by zero or more repetitions of a `message-header`, followed by an empty line (`CRLF`) and finally by an optional `[message-body]` (shown enclosed in square brackets to indicate it is optional). The `start-line`, in turn, consists of either a `request-line` or a `status-line`.

The SIP protocol allows 14 different request methods (listed in Table 1.1) that can be invoked in a `request-line`. Fig. 1.2 shows an example of "good" INVITE request messages which is syntactically correct and semantically meaningful. The reply code is an integer number (in the form of three digit numbers) from 100 to 699 and indicates type of the response. There are 6 classes of responses:


Start Line
Headers
Message Body

```
INVITE sip:user2@server2.com SIP/2.0
Via:SIP/2.0/UDP pc33.server1.com;branch=z9hG4bK776
Max-Forwards : 70
To :  user2 sip:user2@server2.com
From :  user1 <sip: user1@server1.com>;tag=19283017
Call-ID : a84b4c76e66710@pc33.server1.com
CSeq :  3121 INVITE
Contact :  sip:user1@pc33.server1.com

o= user1 2890844526 IN IP4 pc33.server1.com
s = Session SDP
c = IN IP4 157.24.25.137
t = 0 0 :  sip:user1@pc33.server1.com
m = audio 49172 RTP/AVP 0
a = rtpmap:0 PCMU/8000
```

Figure 1.2: Example of a well formed SIP INVITE message

- **Provisional Responses (1xx)**: Range of response codes of this category is 100-199. A provisional response informs its recipient that the associated request was received but the result of the processing is not known yet. Provisional responses are sent only when the processing does not finish immediately. The sender must stop retransmitting the request upon reception of a provisional response. For instance, typically proxy servers send responses to the user agent with code 100 (Trying) when they start processing an SIP INVITE.

- **Successful Responses (2xx)**: A final response is the ultimate response that the originator of the request will ever receive. Therefore final responses express the result of the processing of the associated request. Responses with code from 200 to 299 belong to this category and they indicate that the request was processed successfully and accepted. For instance a 200 OK response is sent when a user accepts invitation to a session (INVITE request).

- **Redirection Responses (3xx)**: Range of response codes of this category is 300-399. 3xx responses are used to redirect a caller. A redirection response gives information about the user's new location or an alternative service that the caller might use to satisfy the call. These responses are usually sent by proxy servers. When a proxy receives a request and does not want or can not process it for any reason, it will send a redirection response to the caller and put another location into the response which the caller might want to try. It can be the location of another proxy or the current location of the callee (from the location database created by a registrar). The caller is then supposed to re-send the request to the new location.

- **Client Failure Responses (4xx)**: Range of response codes of this category is 400-499. These are client error responses. 4xx are negative final responses. A 4xx response means that the problem is on the sender's side. The request couldn't be processed because it contains bad syntax or cannot be fulfilled at that server. For example, typically SIP servers send responses with code 400 (bad request) when they receive SIP INVITE request messages with syntax error from clients.

- **Server Failure Responses (5xx)**: Range of response codes of this category is 500-599. A 5xx means that the problem is on server's side. The request is apparently valid but the server failed to fulfill it. Clients should usually retry the request later.

- **Global Failure Responses (6xx)**: Range of response codes of this category is 600-699. A 6xx reply code means that the request cannot be fulfilled at any server. This response is usually sent by a server that has definitive information about a particular user. For example, user agents usually send a 603 Decline response when the user does not want to participate in the session.

1.2.4 SIP Signaling

SIP is based on the request-response paradigm. The following sequence is a simple example of a call set-up procedure:

- To initiate a session, the caller (or User Agent Client) sends a request with the SIP URI of the called party.
- If the client knows the location of the other party it can send the request directly to their IP address; if not, the client can send it to a locally configured SIP network server. The server will attempt to resolve the called user's location and send the request to them. There are many ways to do this, such as searching the DNS or accessing databases. Alternatively, the server may be a redirect server that may return the called user location to the calling client for it to try directly. During the course of locating a user, one SIP network server can proxy or redirect the call to additional servers until it arrives at one that definitely knows the IP address where the called user can be found.
- Once found, the request is sent to the user and then several options arise. In the simplest case, the user's telephony client receives the request, that is, the user's phone rings. If the user takes the call, the client responds to the invitation with the designated capabilities of the client software and a connection is established. If the user declines the call, the session can be redirected to a voice mail server or to another user.

1.2.5 A Simple Session Establishment Example

Fig. 1.3 shows an example of a call flow from User Agent A to User Agent B. A session is initiated when UAC A sends an INVITE message to the appropriate proxy server indicating that UAC A (caller) wish to communicate/talk with UAC B (callee). Immediately, the proxy server sends a response message (TRYING 100) to UAC A to acknowledge that the INVITE is handled and attempts to resolve the called user's (UAC B) location and sends the request to UAC B. UAC B sends a response (Ringing 180) when his telephone begins to ring. Finally, when UAC B receives the request and picks the call, i.e. UAC B (callee) responds (OK 200 message) to the invitation and a connection is established. Then media streams are exchanged directly between UAC A (caller) and UAC B (callee). The session is ended by an BYE request message to the server from any UAC (A or B) and a reply with an OK 200 message from other UAC.

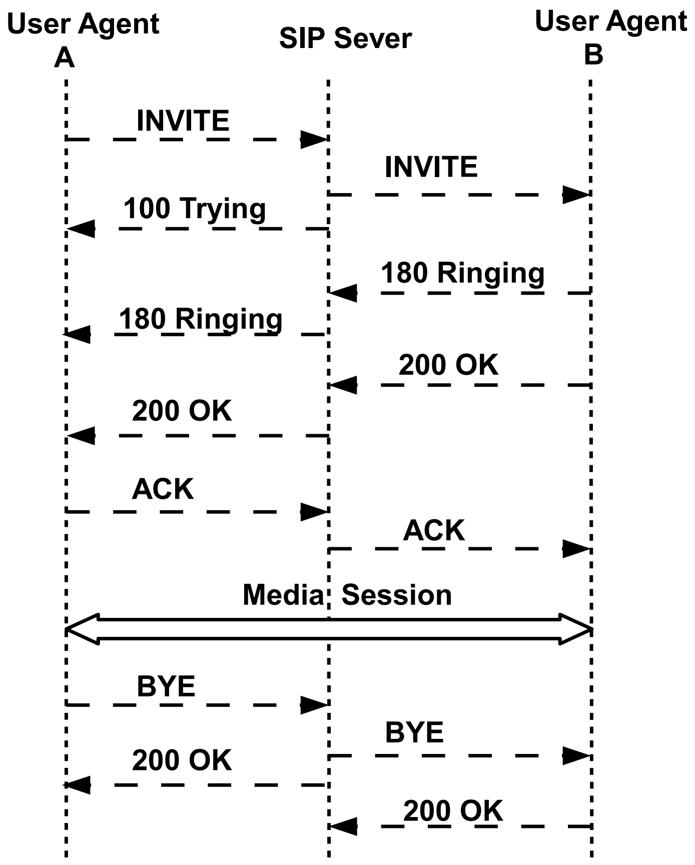


Figure 1.3: Example of a SIP multimedia session

1.2.6 SIP Registration Example

The first step for a user agent to use a SIP-based service is to identify his/her actual location in terms of an IP address. Consequently, the user needs to register the combination of his/her SIP address and current IP address at the SIP registrar responsible for his domain. An example of registration procedure is depicted in Fig. 1.4.

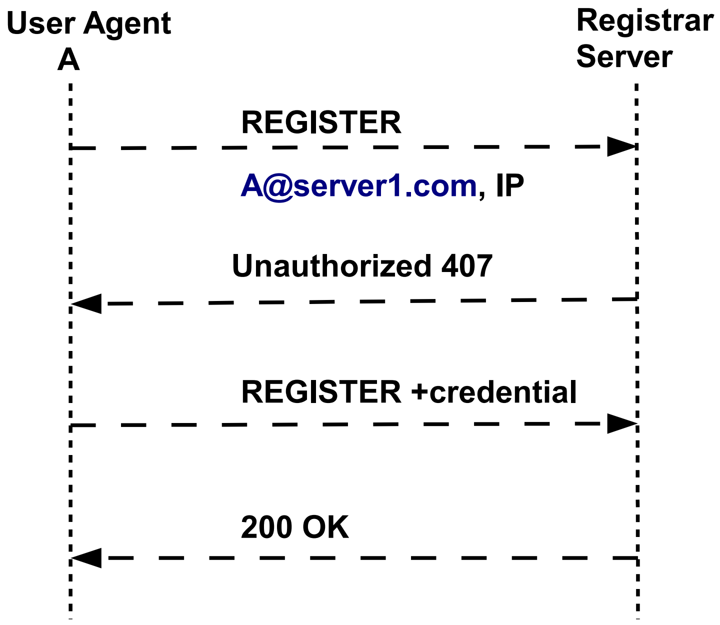


Figure 1.4: Example of SIP Registration procedure

1.2.7 SIP Security

Rosenberg *et al.* [156] in the SIP protocol specification did not define new security mechanism specific to SIP, rather they suggested the reuse of existing security models derived from the HTTP[3] and SMTP[4] space. Later, the IETF has made several improvements that provide protection for the

[3]Hypertext Transfer Protocol (HTTP), RFC 2068, `http://tools.ietf.org/html/rfc2616`
[4]Simple Mail Transfer Protocol, RFC 821, `http://tools.ietf.org/html/rfc821`

SIP based VoIP signaling and media streams. The most relevant recommendations are the use of TLS (Transport Layer Security) [42] to protect SIP signaling and the SRTP[5] to protect the media stream. Authors of [5] proposed functionality supplements to the existing methods of security mechanisms between SIP entities. A secure authentication mechanism is defined in [139] to overcome the inadequacy for cryptographically assuring the identity of the end users that originate SIP requests. A brief discussion on the existing security policies in SIP protocol for ensuring confidentiality, integrity, and authentication is found in [184].

Digest Authentication

SIP provides a stateless and challenge-based mechanism based on authentication in HTTP [60] to provide message authentication and replay protection. It supports both *user-to-user* and *proxy-to-user* authentication. Upon receiving a request message, a proxy server or UA can challenge the initiator of the request to provide assurance of its identity. Only an authorized user is identified and his/her request is processed by the server or UA. While a server can legitimately challenge most SIP requests, *ACK* and *CANCEL* request messages require special handling for authentication.

Transport Layer Security (TLS) protocol

TLS [42] provides transport-layer security over connection-oriented protocols. The protocol is designed to prevent eavesdropping, tampering, or message forgery in client/server applications and can be used to protect communication in SIP based services. TLS is suitable for providing hop-by-hop security between hosts with no pre-existing trust association. For example, UA *A* trusts his local proxy server, which after a certificate exchange decides to trust UA *B*'s local proxy server, which *B* trusts, hence

[5]Secure Real-time Transport Protocol, RFC 3711, `http://www.ietf.org/rfc/rfc3711.txt`

B and A can communicate securely.

IPSec

IPSec is a set of network-layer protocol tools [100] that collectively can be used for securing IP communications by authenticating and encrypting each IP packet of a communication session. IPSec can be used either in *end-to-end* or *hop-by-hop* basic. IPsec can offer confidentiality, integrity, data-origin authentication services by utilizing the Encapsulating Security Payload (ESP)[6] and Authentication Header (AH)[7] protocols. Introducing IPsec in SIP can secure the communication, provided that some sort of trust (e.g., pre-shared keys, certificates) relationship has already existed between the communicating parties. IPSec can provide network layer encryption to secure a VoIP packet stream, but due to the key management issues and problems of integrating with higher level protocols it is not widely used in commercially available UAs.

Secure Multi-purpose Internet Mail Extensions (S/MIME)

Encrypting entire SIP messages end-to-end for the purpose of confidentiality is not appropriate because network intermediaries (like proxy servers) need to view certain header fields in order to route messages correctly. Use of S/MIME [145] can avoid this problem. S/MIME is a standard for public key encryption and signing of MIME data. SIP messages have MIME bodies, therefore S/MIME allows SIP UAs to secure MIME data within SIP. Thus it can provide end-to-end confidentiality and integrity for message bodies without affecting message headers.

[6]IP Encapsulating Security Payload, RFC 4303, `https://tools.ietf.org/html/rfc4303`
[7]IP Authentication Header, RFC 4302, `http://tools.ietf.org/html/rfc4302.html`

Secure Real-time Transport Protocol (or SRTP)

SRTP defines a profile of the Real-time Transport Protocol (RTP) [161] in order to provide encryption, message authentication and integrity, and replay protection to the RTP data and to the control traffic for RTP.

1.3 Security Threats

The last decade has seen a sharp increase in the use of the Internet for voice communications due to the significant cost advantage (e.g., low cost telephony), higher flexibility and richer telephony features (e.g., instant messaging, Internet conferencing rooms, personalized call transfer, etc) than traditional public switched telephone networks (PSTN). Recently, the transition of telephony over IP platforms has entered its final phase: VoIP directly to the home/business users. All major operators in the EU and US are now running SIP-based [101] telephony networks in order to reduce operating costs and to provide modern and richer telephony services.

Though the wide deployment of VoIP has offered numerous advantages for end users and providers, simultaneously has introduced an increasing number of security threats, vulnerabilities and attacks not previously encountered in networks with a closed architecture like the Public Switch Telephone Network (PSTN). The security problem domain of VoIP is considerably larger compared to the traditional PSTN due to several factors:

- **Open Architecture of the Internet:** VoIP/IP Telephony utilizes the Internet architecture, similar to any other data application. However, in an open environment such as the Internet, launching an attack on a telephony server is much simpler as hackers have easy access and much experience. Therefore, VoIP applications inherit a wide set of security weaknesses and threats that are associated with the IP protocol.

- **Convergence of data and voice networks :** In PSTN, security, reliability, and availability are achieved by using a closed networking environment dedicated to a single service. In such services, voice conversations were conducted along dedicated circuits that were either fixed for private networks, or set up temporarily and torn down afterwards in dial-up networks. Either way each call session was shielded both from other voice calls and from other forms of electronic traffic such as data. Moreover, PSTN maintains a separate network for signaling. This made it easier both to protect voice networks from the kind of attacks that data networks were subject to, and also to ensure that the service was highly reliable.

  On the other hand, VoIP is meant to work on an open environment such as the Internet, that means, similar to any other data application it uses the same network for both signaling and data transport. Therefore, due to the lack of a separated and secure signaling and control plane VoIP applications not only inherit the security weaknesses and threats that are associated with the IP protocol but also bring additional security risks.

- **Real Time Communication:** VoIP is a real-time application, it requires timely packet delivery with low latency, jitter, packet loss, and sufficient bandwidth. VoIP is more vulnerable to attack [88, 89, 163] then even from other web/Internet services due to its real time communication constraint.

- **Complex Architecture:** Though the biggest selling points of VoIP are feature-richness, openness and modularity, these are the root of the complexity in architecture of VoIP systems. VoIP technology allows a number of independent implementations and products with distinct configurable parameters and design choices in order to be able to operate in a variety of environments, business settings, and network conditions, resulting high complexity in term of architecture and operation.

- **Use of Open Standards Protocols for Signaling and Media :** As SIP is a text based protocol, it is relatively easy to manipulate and transfer malformed, incorrect, or malicious SIP messages. These messages can cause problems ranging from relatively innocuous disturbances to full blown attacks and frauds. SIP-based applications, appear to be much more sensitive than web services or e-mails to intrusion and mis-functioning.

  Though security and privacy should be mandatory for an IP telephony architecture, most of the attention during the initial design of the SIP-based multimedia architecture has been focused on the possibility of providing new dynamic and powerful services instead of efficient security features and policies. Even, Rosenberg *et al.* [156] did not define new security mechanisms in the SIP protocol specification. Instead, they recommended the adoption of existing security models used over the Internet. It has been found that even with certain existing security mechanisms, SIP protocol is still facing severe security challenges, leaving VoIP applications to security risks. All the elements of the SIP architecture (i.e., proxy-registrar servers and end-user devices) are vulnerable to these kinds of attacks.

As the popularity and use of VoIP application increases, it becomes an highly attractive target for various attacks [102, 199]. A survey of the known/disclosed vulnerabilities of VoIP listed in the Common Vulnerabilities and Exposures (CVE)[8] database is found in [102]. An experiment

[8]Common Vulnerabilities and Exposures, `http://cve.mitre.org/`

with leading deployed VoIP services shows that these services are vulnerable to a number of VoIP exploits that essentially violate the VoIP users' basic trust [199]. Though the motivations are the same for VoIP attackers as they have been for traditional telephone attackers and hackers (e.g., to benefit financially via unauthorized access, to steal identity and information, to gain notoriety by disrupting service, and to disturb users through unwanted calls and to embarrass service providers), success rate of attack generation is noticeably high in the context of VoIP applications. An experiments of SIP end-devices is conducted in [88] to check the robustness of the services. The authors have shown that even the best performing SIP end-devices could not withstand medium level call request flooding attacks. In traditional telecommunication networks, fraud accounts for annual losses at an average of 3%-5% of the operators' revenue and still increasing at a rate of more than 10% yearly [148], while, in VoIP networks, the situation is worse as the possibility and opportunity of occurring fraudulent activity in VoIP is high [13, 141, 185]. Even by hacking a single packet of information from a stream being transmitted over an IP network, *malicious* users can potentially access confidential information or use stolen credentials to access a phone number for fraudulent activity. As a consequence, fraud detection in VoIP is harder than the one in the current telecommunication networks.

With the significant reduction of communication cost VoIP technology has become an attractive target for the hackers. For instance, the transmission of unsolicited calls already exists in the traditional PSTN, where such calls are mostly initiated by telemarketers and *malicious* users for telemarketing and even fraud intention. However, the high cost of PSTN calls compared to email or VoIP communications limits the attractiveness of this form of advertisement for telemarketers. It was found that SIP spam call also known as SPIT (SPam over Internet Telephony) is roughly four orders of magnitude cheaper to send than traditional circuit-based telemarketer calls [155]. Due to this, SPIT which refers to the transmission of unsolicited bulk messages over VoIP has become a urgent problem, similar to that of email spam. Many attacks that are launched against VoIP users and networks are not successfully launched against the traditional circuit-switched telephone network. Particularly, in PSTN it is considered very difficult to launch any kind of attack without having any physical access to the network. This is not the case for VoIP based Internet services, since a vulnerability in the service/protocol can be exploited easily by utilizing various VoIP security testing and attack generation tools. For example, eavesdropping in the context of VoIP is easier than PSTN. Eavesdropping in VoIP is somewhat different from the traditional eavesdropping in data networks, but the general concept remains the same. It requires intercepting the signaling and associated media streams of a conversation. In case of traditional telephony an attacker cannot easily access and install a tap on a telephone pair outside victim's house as it implies more exposure. On the other hand, IP eavesdropping can be done from the comfort of attacker's laptop as long as he/she posses the tools and expertise to carry out the attack successfully (e.g., to sniff the media transmission and transform it in an audio file). Again, the man-in-the-middle (MITM) attack is also considered one of the most serious threats to the security and trust of existing VoIP protocols and systems [31, 210]. For example, the MITM who is in the VoIP signaling and/or media path can easily wiretap, divert and even hijack selected VoIP calls by tampering with the VoIP signaling and/or media traffic.

1.4 Motivation of the Thesis

With the increasing popularity of VoIP services, the availability, confidentiality, and integrity of VoIP communication has become increasingly compromised by remote intrusions. Though SIP-based VoIP has become a popular alternative to traditional PSTN due to its flexibility and feature richness, it is more vulnerable to attack than the traditional architecture and even from other web/internet services. Ensuring security with strong access control and encryption policies is hardly scalable in the highly open and dynamic architecture of the Internet. To this end, the IETF has made several improvements that provide protection for the VoIP signaling and media streams. Due to the overheads of implementation and performance these security mechanisms have not been fully implemented and deployed in current VoIP applications. Moreover, existing security solutions are not efficient to cope with the increasing sophisticated large-scale attacks and malwares situations in SIP based services. In such a scenario, analysis tools taking into account service semantics and troubleshooting VoIP systems based on SIP are of paramount importance for network administrators.

SIP seems to be quickly gaining in acceptance as the underlying protocol for many VoIP applications. Since SIP is so widespread and ubiquitous, great care should be taken to ensure its resilience and security in presence of incorrect, malformed or malicious messages that could disrupt or perhaps even block entirely the flow of messages across a telecommunication network. Session-based applications, and conversational ones in particular, appear to be much more sensitive than web services or e-mails to intrusion, mis-functioning and even incorrect interpretation of the standard, if not for else, because their real-time nature prevents off-line inspection or semantic analysis of the content.

Though the extensibility and features' richness of SIP protocol determined its success, but also decreed the drawback of making SIP the most complex IETF standard ever. This, together with the additional extensions added by other standards organizations and consortiums, have turned SIP in a protocol hard to debug and troubleshoot. As if this was not enough, telecommunications operators are using today SIP to implement many different services resulting in a complex interdependency across numerous network and service elements. All these factors give raise to an enormous number of possible states which makes conformance testing not feasible (as for most Internet protocols), so that in the end, many different implementations and "dialects" of SIP coexist in the network, and they must be accounted for in devising methods for analyzing SIP traffic and protect servers and applications. Moreover, the fact outlined above, changes non marginally the rules of the game in that the characteristics of voice traffic do not match any more the traditional Erlang models [30,37,169], so a new understanding of the traffic characteristics for SIP-based services is badly needed.

Analysis and protection of SIP based services has thus become an active area of research, as well as the specific actions that can protect the protocol, or the firewalling techniques specifically devised for it. Hence, this thesis focuses on the analysis and protection of services based on this protocol. We note that most, if not all, of this thesis deals with SIP-based VoIP application though an extension of the security framework proposed in this thesis is possible for other SIP based multimedia services.

Due to the open architecture of the Internet, heterogeneous environment and real time communication constraint of SIP based VoIP networks, developing efficient, flexible and adaptive security oriented approaches is a severe challenge. For example, inspite of conceptual similarities between spam email and calls (e.g., SPIT), many effective email spam prevention methods are not applicable at all in anti-SPIT system. Indeed, both email spam and SPIT have the same underlying motive, i.e., financial profit. Attackers, in both cases, use the Internet as the means to target users and deliver their attacks (messages). Additionally, spam and SPIT share similar kinds of attack ideas and methods, such as automatic generation of bulk messages for cost reduction, impersonation of end users' addresses, harvesting addresses, as well as dictionary attacks. Considering above similarities, researchers adopt and apply effective anti-email-spam techniques for managing SPIT. Inspite of the similarities, SPIT is a much bigger threat for users than email spam due to the different nature of the SPAM and SPIT. The main difference between SPAM and SPIT is that spam e-mail is transmitted asynchronously, while SPIT uses real-time communication. A SPIT call makes the phone ringing and disturbs the callee. On the other hand, email SPAM can be queued in the email program of the receiving user without disturbing the user until he looks at it. Unsolicited communications (such as, SPIT) are, from a signaling point of view, technically correct transactions. It is not possible to distinguish from the INVITE message if such a transaction is SPIT or not. From a technical point of view the challenge is even more complicated since a spam call must be identified before call establishment, while, the content is not available to help the detection until the phone rings (disturbing the user) and the callee answers the call.

The research community has been investigating the best ways of attack detection and protection for the VoIP services for the last decade. Intrusion Detection Systems (IDSs) [6, 124, 133] are considered an effective technology to protect target systems and networks against malicious activities. IDS is a well-known concept in the field of security of IP based networks as a second line of defense behind standard security mechanisms (e.g., authentication, access control and cryptography) to detect violations of the security policies. Intrusion detection is the process of monitoring network traffic (e.g., the network and application protocol activity) for particular network segments or devices to identify suspicious activity such as malware (e.g., worms, spyware), *malicious* users gaining unauthorized access to services, and authorized users of services who misuse their privileges or attempt to gain additional unauthorized privileges. Traditionally, IDSs are broadly classified into two groups based on their working principles: (i) *misuse-detection* system which monitors activity with precise descriptions of known *malicious* behavior, and, (ii) *anomaly-detection* system (ADS) which identifies intrusion considering the notion of normal activity of the network. Though *misuse-detection* systems ensures higher performance for known attacks, they are not effective for detection of new attacks and requires constant update of attack information. While, anomaly-based intrusion detection techniques [62, 112, 187] are widely used for detection of both known and novel attacks. ADSs are based on the beliefs that an intruder's behavior will be noticeably different from that of a legitimate user. Anomaly-based IDS detects anomalies by searching the deviation from normal activities due to deferent attacks, misconfiguration of network components, network failures etc.

Research on ADSs is gaining importance due to the increasing threats toward VoIP systems. Research attempts focusing ADSs for multimedia/VoIP applications is still in the early phase as VoIP networks are still relatively new in comparison to IP networks. In the context of VoIP domain, special attention should be given in the design and deployment of ADSs as most of the ADSs dedicated for detecting traffic in the TCP/IP traffic may not be efficient for VoIP network due to complex and heterogeneous architecture of VoIP systems and real-time nature of the VoIP application.

In this thesis, we focus on an in-depth analysis of the SIP-based VoIP services to obtain a solid insight of the network and service behavior. This information helps us to design and implement of a filtering architecture for SIP based VoIP networks where the incoming network data is captured and inspected on-line for detecting on-going attacks or malicious activities.

1.5 Structure of the Thesis and Contributions

A general overview of how the problems we tackle in this thesis fit together, as well as how they lead to the organization of our text, is depicted in this section.

Chapter 2 reviews the state-of-the-art in vulnerability of SIP-based VoIP services and countermeasures that has been proposed so far in the literature to tackle the security risks in the context of VoIP network.

Chapter 3 outlines the methodological approach we follow for analyzing and protection of SIP based services.

Chapter 4 introduces a knowledge base defined as “VoIP-Onto”, a formal conceptual model of VoIP threat domain considering the semantic relationship among security threats, attacks and countermeasures, with the aim to exchange a common vocabulary about the security related information of the VoIP domain. “VoIP-Onto” can be used as a general vocabulary, road map, and extensible dictionary for sharing domain knowledge about taxonomy of VoIP attack and guidelines of counter measures. This contribution is summarized in Chapter 4.

Chapter 5 points out the issues and challenges of collecting real-world VoIP traffic for the purpose of testing security monitoring and controlling tools. The process we adopt for capturing and anonymization of real world SIP traffic is discussed in Section 5.1. This Chapter also presents our research work in generation of synthetic traffic, introducing “VoIPTG” which is a flexible and generic traffic simulator and “SIP-Msg-Gen” which is a SIP message fuzzer. “VoIPTG” is capable to imitate real-world VoIP traffic (e.g., call detail records) following various possible models of VoIP systems with a special attention to characterize the sophisticated behavior of VoIP users. This contribution is summarized in Section 5.3. “SIP-Msg-Gen” is a state-full and context aware simulator which is capable of generating both “good” and “bad” SIP messages. This contribution is outlined in Section 5.2.

Chapter 6 presents our research work in conformance and security analysis of SIP messages, proposing a multi-stage filtering architecture to address the challenges of filtering “erroneous” SIP messages with different structure, contents and timing. The first stage of the multi-stage filtering system is a straightforward *lexical analyzer* to weed out the SIP messages with *syntax error*, while, the remaining three filters follow supervised machine learning techniques to detect harmful SIP messages with *semantic* errors by learning from previous experiences. The experimental results of this filtering system look very promising and are presented in Section 6.2.

Chapter 7 summarizes the analysis of VoIP traffic with the aim to understand the normal behavior of the network and users. The notion of *social network* is applied in the context of SIP based VoIP network, where *social networks* of VoIP users are built based on their telephone records. Then, Social Network Analysis (SNA) techniques are applied on these *social network* of VoIP users to explore their social behavioral patterns. Section 7.3 presents the impact and use of the knowledge about the social interaction pattern of VoIP users in problem diagnosis, intruders detection, and security protection of IP telephony.

Finally, concluding remarks and insights about future possible research directions are provided in Chapter 8.

Chapter 2

State of the Art

This chapter reports various security issues and challenges in VoIP infrastructure, and presents a discussion on security solutions and countermeasures adopted by researchers and vendors to ensure secure deployment and operation of VoIP applications.

2.1 VoIP Attack Taxonomy

Despite the use of security mechanisms, SIP based VoIP services are still subject to certain vulnerabilities [63]. A good starting point for designing and deployment of effective countermeasures is to understand different vulnerabilities and security threats to VoIP domain and how they are exploited to launch different attacks. This section provides a taxonomy of the different types of security threats to VoIP deployments, services, and end users.

There are various categorizations and taxonomies of VoIP specific attacks exist in literature based on the different viewpoints and consideration of researchers. The Voice over IP Security Alliance (VoIPSA) [206] has created an enormous classification for VoIP threats and attacks, which is an important contribution to understand the associated threats. But that taxonomy is a very complete one (as it takes into account also threats not caused by VoIP-specific technical reasons) for practical VoIP security analysis. A similar taxonomy is presented in IETF draft [131] where threats only caused by VoIP-specific technical reasons are considered. Author of [101] examines the current state of affairs on VoIP security through a survey of 221 known/disclosed security vulnerabilities in bug-tracking databases and presents a comprehensive survey of the state of the art in VoIP security research covering 245 related research papers. This book supports the taxonomy described in the IETF draft [131].

A different taxonomy is defined in [24, 121] where each security threat is classified regarding its compromise on the information security attributes, i.e., *Confidentiality*, *Integrity* and *Availability* (CIA). In the telephony domain, *confidentiality* threats generally refer to the exposure of the communication (e.g, content of the conversation, information about call such as telephone numbers dialed, call duration, etc) between two parties, *integrity* threat refers to the unauthorized access and modification to the communication services (e.g., identity of the caller, the message, the identity of the recipient, or the call record logs). *Availability* threat indicates compromising the availability of the service or degrading the quality level of such resources. Though classification of attacks based on *CIA* are widely accepted, according to [184] this type of categorization is not suitable for defining a taxonomy due to the fact that a taxonomy should be unique and many vulnerabilities should not belong to several categories at the same time.

In this work, we follow the VoIP attack taxonomy proposed in the IETF draft [131] where the security threats can be categorized in four areas as discussed in the following subsections. For the time being, we limit the scope of our discussion on vulnerabilities, threats and attacks of VoIP domain specific to SIP protocol.

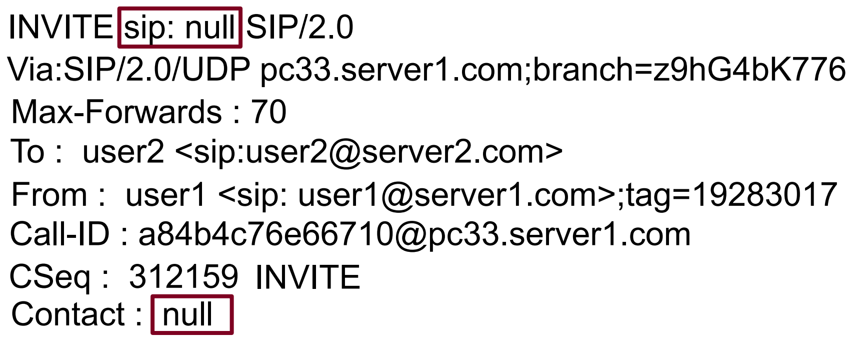
2.1.1 Interruption of Service Threats

Attacks of this category target network bandwidth, server capacity or service quality. This category mainly includes the Denial of Service (DoS) attacks that aim at denying or degrading a legitimate user's access to a service or network resource, or at bringing down the servers offering such services. An overview of different types of possible DoS attacks in SIP based VoIP services is explained in [173]. VoIP is more susceptible to DoS attacks than regular Internet services due to the real time transmission of the messages [89]. DoS attacks can be launched in distributed fashion and directed to different entities (e.g., servers and end-users) depending on the attackers' intension. A list of the well-known DoS attacks in SIP based VoIP services is found below:

- **Fuzzy SIP Messages -**
  Fuzzy SIP messages are all those messages that do not belong to a valid, correct and legitimate SIP session. They can be generated when SIP protocol implementations or applications do not fully comply with the standards or they contain errors in the implementation code. In addition, attackers can intelligently manipulate SIP messages to take advantage of the existing security problems in the target system, or to exploit SIP weak points.

  Due to the text format of messages of the SIP protocol, "fuzzy" messages can be easily generated. As a matter of fact, there are endless ways to manipulate SIP messages: they can be just *malformed* in the sense that they do not comply with the grammar [156] prescribed by the protocol or they could be syntactically well formed but containing *semantic* errors. An example of *malformed* SIP message is found in Fig. 2.1 in which a valid receiver address is missing. Generally, the server receiving this message should simply respond with a "Bad

Request" indication and discard the message; however, under certain circumstances, the server may then fail to process further messages or it could get stuck altogether.

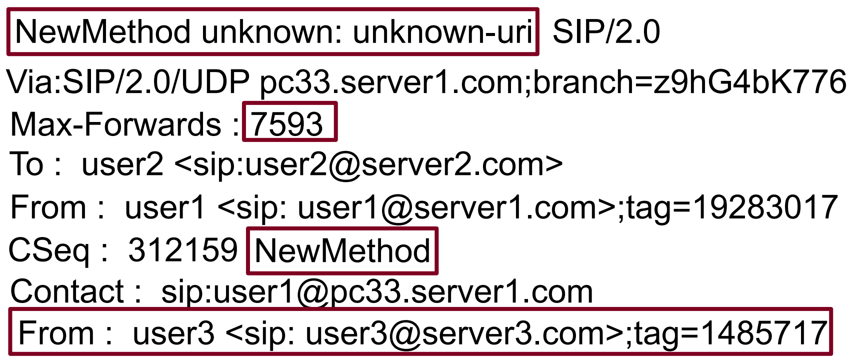


```
INVITE sip: null SIP/2.0
Via:SIP/2.0/UDP pc33.server1.com;branch=z9hG4bK776
Max-Forwards : 70
To :  user2 <sip:user2@server2.com>
From :  user1 <sip: user1@server1.com>;tag=19283017
Call-ID : a84b4c76e66710@pc33.server1.com
CSeq :  312159  INVITE
Contact : null
```

Figure 2.1: An example of a SIP message with syntax error

SIP messages that are syntactically well-formed may still contain semantic errors. These messages may have no meaning, cannot be interpreted, are ambiguous, lead to a deadlock, etc. For example, Fig. 2.2 shows a SIP message with an unknown request method, an unknown URI scheme in the Request-URI, missing mandatory header field (Call-ID), scalar fields with overlarge values and multiple entries for headers that ought to be specified only once (like the *From* header). The message is syntactically valid but a server receiving this message will fail to process it and may perform time-consuming analyses to determine the request message type and acquire the information that are necessary to route and process the request.

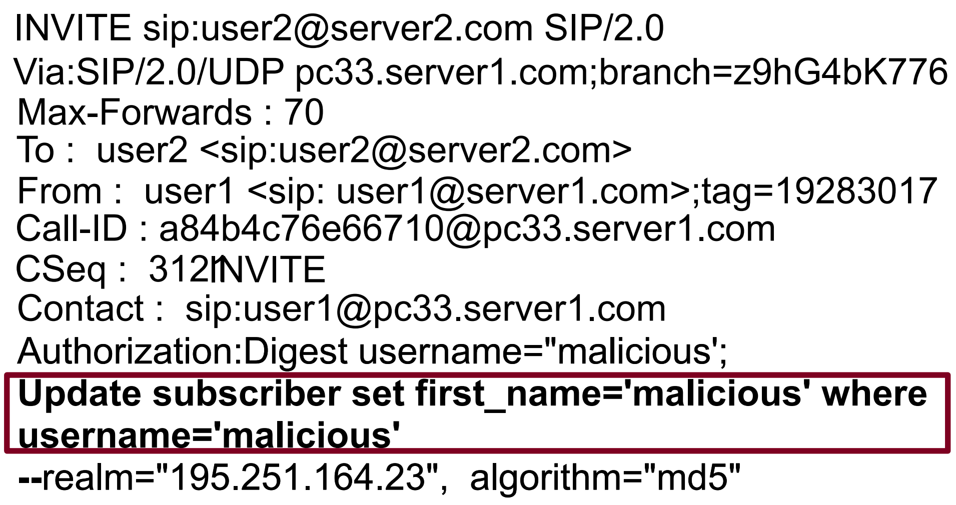
In addition, attackers can manipulate SIP messages by entering meaningless or wrong information in various fields of a message to take advantage of existing security problems in the target system or to carry out different kinds of attacks. Again, a *malicious* user can inject SQL code into a well-formed SIP message that may damage the system (e.g., updating or deleting entry in table containing user account



```
NewMethod unknown: unknown-uri SIP/2.0
Via:SIP/2.0/UDP pc33.server1.com;branch=z9hG4bK776
Max-Forwards : 7593
To :  user2 <sip:user2@server2.com>
From :  user1 <sip: user1@server1.com>;tag=19283017
CSeq :  312159 NewMethod
Contact :  sip:user1@pc33.server1.com
From :  user3 <sip: user3@server3.com>;tag=1485717
```

Figure 2.2: An example of syntactically well-formed SIP message with semantic error

information) when executed on the server [64]. Fig. 2.3 shows an example of SIP message with injected SQL code by a malicious user with the intention of gaining unauthorized access of the SIPs proxy database (e.g. the registration database).



```
INVITE sip:user2@server2.com SIP/2.0
Via:SIP/2.0/UDP pc33.server1.com;branch=z9hG4bK776
Max-Forwards : 70
To :  user2 <sip:user2@server2.com>
From :  user1 <sip: user1@server1.com>;tag=19283017
Call-ID : a84b4c76e66710@pc33.server1.com
CSeq :  312INVITE
Contact :  sip:user1@pc33.server1.com
Authorization:Digest username="malicious';
Update subscriber set first_name='malicious' where
username='malicious'
--realm="195.251.164.23",  algorithm="md5"
```

Figure 2.3: An example of a SIP message tampering by injecting SQL code

- **Flooding Attack** -
  A common DoS attack scenario in SIP based VoIP is message flooding attack where *malicious* users send huge number of INVITE/REGISTER request messages to the SIP server (or any SIP entity), thus, the target

system becomes so busy processing call requests from the malicious users that it is unable to process legitimate call requests. Legitimate packets will either be ignored or processed so slowly that the VoIP service is unusable.

Flooding attacks can also occur with the existence of authentication mechanisms. In such scenario, after receiving INVITE/REGISTER requests messages the proxy/registrar responds with a challenge and waits for the request to be send again with the proper authentication credentials. The attacker can send INVITE/REGISTER messages and then stop handshaking process. Two types of flooding attack scenarios are usually observed:

– **INVITE Flooding Attack**:
  Malicious users generate INVITE Flood attack by sending a large number of SIP INVITE messages to a SIP server. With this form of attack, the SIP server remains so busy processing these false INVITE messages from the attacker, that it will not be able to process call requests from legitimate users. Fig. 2.4 shows an example of SIP INVITE message flooding attack scenario.

– **REGISTER Flooding Attack**:
  One of the cardinal network elements in SIP based VoIP infrastructure is the Registrar server that accepts the SIP registration requests from user agents. A SIP REGISTER request adds a new binding between a user's SIP address and one or more contact addresses (currently IP addresses) so that the user can utilize the provided telephony service. When an attacker manages to paralyze the registrar (e.g., by sending numerous bogus registration requests), it can easily cause a DoS attack. A *malicious* user attacks a Registrar server by sending a large number of registration

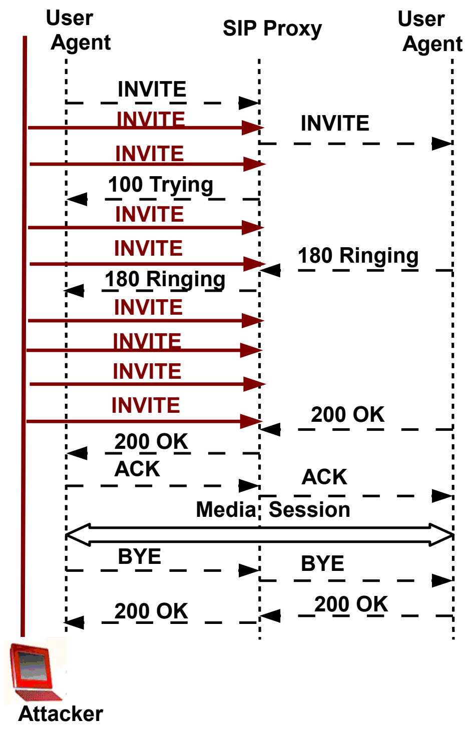


Figure 2.4: An example of a SIP INVITE message flooding attack

request (e.g., SIP REGISTER messages) with the aim to guess legitimate users' password or to degrade and delay the registration process of the server. This situation can be avoided by blocking messages coming from unknown origins.

Fig. 2.5 depicts such an attack scenario against a SIP registrar server.

- **Call Hijacking** - Call Hijacking happens when the attacker uses SIP messages (e.g., 301 Moved Temporarily) in order to hijack an existing call towards other proxy/endpoint, it is needed that the attacker replicates the proper SIP header for the hijacking to be successful.

- **Call Teardown** - Call Teardown happens when the attacker uses SIP messages (e.g. CANCEL/BYE) order to tear down an existing call by replicating the proper SIP header of the existing call.

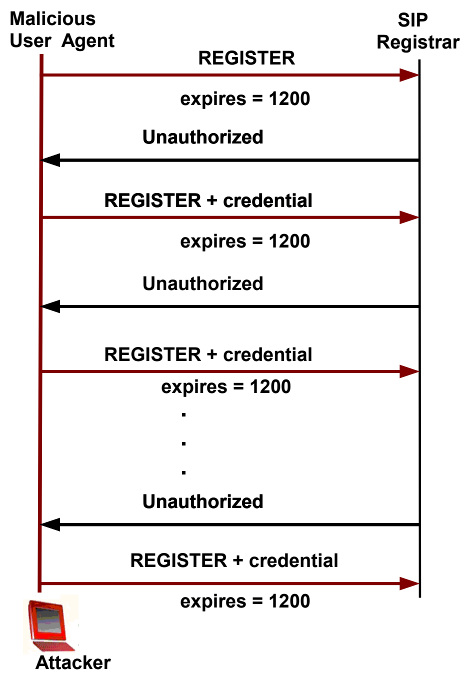


Figure 2.5: An example of flooding attack against SIP Register Server

– **CANCEL Denial of Service Attack:**

The CANCEL attack refers to the unauthorized tear down of an non-established call between two user agents. The CANCEL request method is used to terminate pending searches or call attempts. Specifically, it asks the UAS to cease the request processing and generate an error response to the request. It can be generated either by a user agent or the proxy servers while the call establishment is in progress.

Fig. 2.6 depicts an CANCEL dos attack scenario. Here, in order to initiate a new call user agent A sent an INVITE request message to user agent B. Upon receiving the INVITE message B's SIP proxy receive sends back an "100 Trying" response indicating that it is processing the call request. When B's phone starts ringing, a "180 Ringing" is sent back to notify A that the call establishment is in progress. At this time, an attacker can easily send

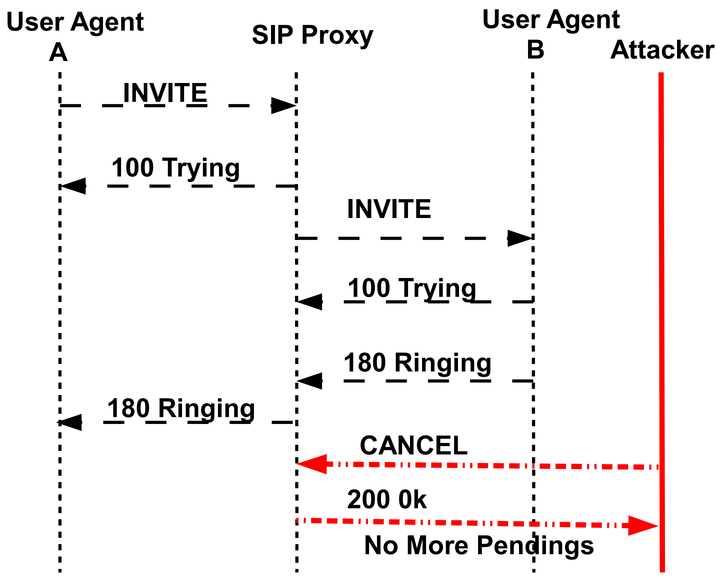


Figure 2.6: An example of CANCEL dos attack

a spoofed CANCEL message to the server. Without proper authentication, the server cannot differentiate the spoofed CANCEL message from the genuine one and considers that the CANCEL request is coming either from user agent *A* or *B*. Upon receiving the CANCEL message the server tear down the call establishment process between user agents *A* and *B*.

– **BYE Denial of Service Attack:**
An established media session between two user agents is terminated upon receiving a BYE message from either of them. It is an end-to-end message sent by user agents participating in the session. The BYE attack refers to the unauthorized tear down of an established call between user agents. Fig. 2.7 depicts an BYE dos attack scenario. Here, a call is established between UA *A* and UA *B* with three-way handshake process (INVITE-200 OK- ACK message). Suddenly, a malicious user sends a BYE message to

either UAs, *A* or *B*. The receiving UA may not differentiate the spoofed BYE message from a genuine one and will prematurely teardown the established call assuming that it is requested by the partner UA.

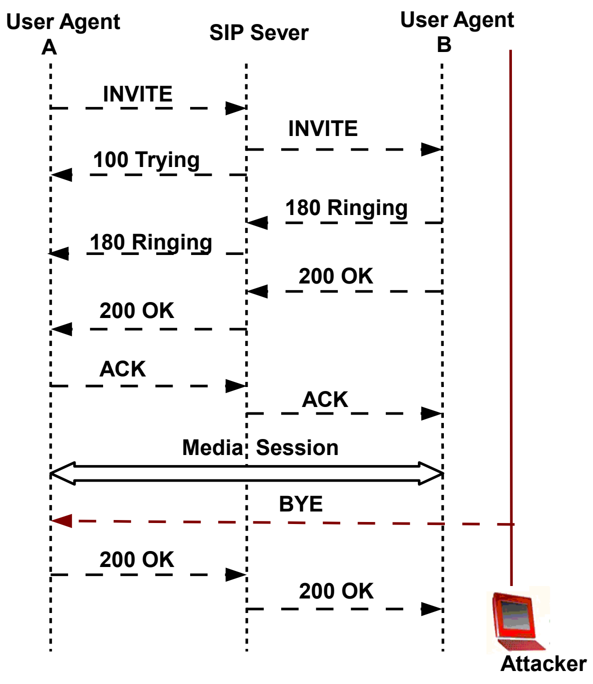


Figure 2.7: An example of BYE dos attack

2.1.2 Abuse of Service Threats

Service abuse threats and attacks cover the improper use of VoIP services, especially (but not exclusively) in those situations where such services are offered in a commercial setting. Examples of such threats include fraud and billing avoidance which can be simply described as the use of VoIP telephony services without any intention of paying.

- **Toll Fraud** - The term "Toll Fraud" is used if a person generates costs (toll) by using a hijacked extension. In this case, an attacker can impersonate a valid user/IP phone and use the VoIP network for making free long distance calls. Example of toll fraud is the hacking of a company's phone system by a third party.

This kind of fraud has certain characteristics that make it particularly attractive to fraudsters. The main one is that the danger of localization is small. This is because all actions are performed from a distance, which in con- junction with the mess topology and the size of networks makes the process of localization time-consuming and expensive. Additionally, no particularly sophisticated equipment is needed. The simple knowledge of an access code, which can be acquired even with methods of social engineering, makes the implementation of fraud feasible.

- **Session Replay** - In this attack the past session of another user can be replayed by the attacker to have access to the unauthorized resources.

2.1.3 Interception and Modification Threats

Interception, and modification threats cover situations where an adversary unlawfully and without authorization can listen or modify the signaling data or the content of a VoIP session. Eavesdropping and Man-in-the-Middle attacks can be categorized into this area. Examples of such attacks include Traffic capture, Number harvesting, Call black holing, Call rerouting, Conversation alteration, Conversation degrading etc.

2.1.4 Social Threats

Social threats are attacks ranging from the false representation of information together with generation of unsolicited communications with the aim to simply disturb users and even to steal private information. Social threats are aimed directly against humans, these threats are classified as social since the term *unsolicited* is strictly bound to user-specification preferences. This makes this kind of attack difficult to identify. Examples of social attacks are:

- **SPIT** - SPam over Internet Telephony (SPIT) commonly refers to the transmission of unsolicited voice messages with telemarketing, phishing or fraud goals. A SIP call SPAM application is easy to write, most of the cases spam calls are generated by machines (bot-nets) programmed. For example, a SIP UA can initiate a large number of calls in parallel. If a call connects, the SPAM application generates an ACK and proceeds to play out a recorded announcement, and then it terminates the call. This kind of application can be built entirely in software, using readily available off-the-shelf software components. It can run on a low-end PC and requires no special expertise to execute [155].
- **VoIP Pishing/ Vishing** - The term *vishing* refers to the *phishing* in VoIP domain. In general, phishing is about masquerading as a trustworthy third party in an attempt to acquire confidential information from victims. In the context of VoIP, an attacker presents false or misleading credentials by impersonating a valid user/enterprise over the phone in order to gain access to personal and financial information. Again, the attacker can make the callee dialing expensive numbers in order to get the promised prize or to collect his/her personal data by redirecting the user towards an Interactive Voice Responder (IVR) pretended to be trusted.

2.2 VoIP Security Testing Tool

A number of competing independent implementations of SIP based services with distinct configurable parameters and design choices coexist in the network. Not all of them are completely stable and are susceptible to attacks where malicious uses try to discover possible security flaws by exploiting the weakness of the applications. These applications should be undergone a full security assessment in the development phase in order to discover vulnerabilities and flaws in implementation and configuration.

Testing of SIP based services is complicated due to the complex and stateful nature of the protocol; however, different approaches have been adopted by various researchers and vendors to test basic functionality and to verify proper protocol implementation in SIP based equipment. VoIPSA provides a categorical list of the available commercial and open source VoIP security testing tools[2]. Authors of [121] make an attempt to evaluate the capabilities of few available VoIP security tools to detect potential security threats.

A set of SIP messages defined as "SIP Torture Test Messages" is released as an IETF-draft [177]. This draft does not contain a comprehensive list of unusual, but valid, messages, instead, it focuses on the scenarios that should be handled properly before deployment of SIP services. It defines 42 valid and invalid messages, describes them and gives directions on how the SIP application should react.

Fuzzing, a well-known test technique to find vulnerabilities in different applications, is also used in SIP-based VoIP application for security testing. In the context of SIP based services, fuzzing refers to the automated way to test the robustness and security of the implemented applications using SIP protocol by providing invalid, unexpected or random data to the SIP traffic. "Snooze" [11] is a tool for building flexible, security-oriented and stateful network protocol fuzzers to effiectively identify security flaws in network protocol implementations. The authors build a prototype of this

[2]VoIP Security Tool List, VoIP Security Alliance (VOIPSA),
`http://www.voipsa.org/Resources/tools.php`

tool to perform fuzzing of SIP based applications. A framework for SIP security testing is proposed in [178]. This tool consists of (i) a fuzzing unit to craft malicious SIP messages, and (ii) a DoS attack simulator to spoof any given range of usernames and IP addresses. “KiF”, a stateful fuzzer, is proposed in [1] to detect failures and bugs in the implementations of SIP based services. SIP messages are generated with the help of a syntax fuzzer and the state protocol evaluator. The syntax fuzzer is uses the domain knowledge of SIP which is represented by the Context-Free grammar ABNF, while, the state machine is used to model the transitions in the system.A state-aware for application layer protocol is proposed in [107]. The functionality relies mostly on user defined scenarios which are specified in XML.

To test the robustness of SIP implementation, the authors of [201] prepared a test material and carried out tests against a sample set of existing implementations. The test suit consists of crafted SIP messages containing exceptional elements to provoke undesired behavior in the implementation. An exceptional element is an input that might not have been considered properly when implementing the application. The robustness of a SIP implementation is measured here by the ability of a application to withstand exceptional input and stressful conditions.

A few attack simulators are available to aid the security and conformance testing of SIP based VoIP services. A list[1] of VoIP security related resources contains the security tools and attack simulators that are demonstrated by the author of [48] to test security of VoIP applications. “VoIP bots” [128]) is a VoIP security tool supporting a wide set of attacks ranging from SPIT to DDoS. It is created to show how attacks can be launched against VoIP/SIP services and users in a remotely and distributed manner.

[1]Hacking Exposed VoIP, `http://www.hackingvoip.com/tools.html`

"Sipvicious"[2] is an open source attack simulator containins several small tools; (i) a sip scanner, (ii) a tool to find active SIP accounts with REGISTER messages, and (iii) an online password cracker for SIP PBX. "SIPp"[3] is a open Source traffic generator for the SIP protocol. It generates traffic by repeating call flow scenarios described in custom XML, hence it is convenient for benchmarking and assessing stress conditions at SIP servers. It includes a few basic SipStone user agent scenarios (UAC - Client and UAS - Server) and establishes and releases multiple calls with the INVITE and BYE methods. " SIPp-DD" is a flooding attack simulation tool for imitating real life distributed denial of service attack. It modifies and extends the open source traffic generator "SIPp" to simulate attack.

A tool for customizing and generating synthetic VoIP traffic within controlled environments is proposed in [193]. This tool is capable of simulating customize VoIP traffic; the normal traffic is generated by profiled emulated users based on a social model. The attack traffic is based on currently available VoIP assessment tools.

"SIPr"[4] is an open source SIP application testing framework. It can create complex and converged SIP application call flows.

2.3 Attack Detection & Protection of SIP Services

The security issue of SIP based services becomes apparent with the increase of the popularity of these services. Though combination of strong access control and encryption policies can be a good approach to protect attacks against SIP based services, ensuring these policies is hardly scalable in the highly open and dynamic architecture of the Internet. The IETF has made several improvements that provide protection for the VoIP signaling and

[2] "Sipvicious", `https://code.google.com/p/sipvicious/`

[3] "SIPp",`http://sipp.sourceforge.net/`

[4] http://www.agnity.com/sipper/index.php

media streams. Due to the overheads of implementation and performance these security mechanisms have not been fully implemented and deployed in current VoIP applications. Moreover, existing security solutions are not efficient to cope with the increasing sophisticated large-scale attacks and malwares situations in SIP based services. As a result, intrusion detection systems (IDS) have become an indispensable component of security infrastructure to detect intrusion or anomalous behavior that passes the existing security policies.

Due to the different nature and content of attacks, their detection methods are also different. Thus, researchers and vendors are investigating various approaches of assuring different security aspects [20,86,97,127,129,202]. This section focuses on papers that highlight the different approaches adopted by various researchers for VoIP traffic analysis and intrusion detection in the context of SIP. A survey of VoIP security domain is found in [101] covering 245 VoIP security research papers and books.

2.3.1 Defense against Interruption of Service

Interruption of services such as DoS attacks are serious threats for SIP VoIP infrastructures [109]. An extensive overview of the recent attempts to detect SIP-based Dos attacks is found in [45, 63].

Detection of SIP malformed message

A specific "signature" considering the syntax of well-formed SIP message defined in the IETF standard of SIP protocol RFC 3261 [156] is defined in [65] and [113]. Any message that does not comply with that "signature" is considered as malformed and discarded. An intrusion detection system for SIP-based VoIP system utilizing rule matching algorithm and state transition models is proposed in [168]. Authors of [65] and [168] point out that the rules of SIP messages defined in RFC 3261 [156] cannot cover all kinds of malformed SIP messages (for example, it does not define any range for scalar fields) and hence [168] has proposed an extension to RFC 3261 by introducing additional rules to make it (more) secure. Generally, such signature-based intrusion/anomaly detection systems work well when no entirely new, uncataloged attacks occur and the "attack signatures" database is not huge. However, handling of intrusion/anomalous detection problem in the context of SIP with a table-drive approach (rule/signature database) is destined to run up against the combinatorial explosion, as there are endless ways of forming a malformed or malicious message. Again, there are multiple ways of structuring a correct SIP message. Other researchers have proposed machine learning techniques for SIP messages analysis, like in [81, 129, 144, 152] where the anomalous content is identified by parsing SIP messages. A self-learning anomaly detection system is proposed in [152] which emphasizes the detection of unknown and novel attacks. Here, incoming SIP messages are mapped into feature spaces and the anomaly detection model is trained using normal/well-formed SIP traffic. Anomalous messages are identified as those whose Euclidean distance from those in the model of normality is higher than a given threshold. [81], however, reports that classifiers based on Euclidean distance computation do not produce adequate results for well-crafted malicious messages that differ very slightly from normal messages and hence its authors suggest a classifier based on Levenshtein distance [75] to measure the similarity between good and bad SIP messages.

Detection of SIP flooding attack

A two-layer attack detection and protection architecture for SIP based VoIP services is proposed in [47] to handle different types of attacks, including request flooding and malformed message sending. The authors configure "Snort" [154], a widely used open source network intrusion detection and prevention system, with a set of rules for detection of flooding attacks on SIP infrastructures. [137, 207] implement a large-scale, rule-based SIP-aware application-layer-firewall capable of detecting and mitigating SIP-based Denial-of-Service (DoS) attacks at the signaling and media levels.

Different statistical methods are used in [76, 149, 164, 182] to model the normal traffic flow of a network. If the observed behaviors significantly deviate from the normal traffic flow then it is suspected that the system is under flooding attack. Authors of [76] analyze SIP traffic flow to extract features describing the statistics and behavior of the flow and uses them for detecting distributed dos attacks. *Change-point* method is used in [149,151,208] to detect changes in network traffic which leads to flooding attacks. The *change-point* detection method employs a statistical analysis on the correlation between the number of connection establishment attempts and the completed handshakes to detect very subtle traffic changes from SIP protocol behavior. Authors of [164] and [182] use Hellinger distance (HD) to detect flooding attacks. Hellinger distance [132], a metric that quantifies the deviation between two probability measures, detects flooding attack scenario by observing the difference in probability distributions from normal traffic. [164] considers the normal distribution of the call establishment (INVITE-200 OK-ACK SIP messages) and teardown (BYE/CANCEL-200 OK SIP messages) in a system. Here, the attack threshold i.e. the calculated Hellinger distance value is dynamically adapted based on previous monitored network parameters.

Protocol state machines based IDS are proposed in [29, 46, 163, 167] where state machines provide the protocol design in term of desirable and undesirable protocol states and state transitions, and classify a deviation from normal behavior as a suspicious attack. To detect INVITE flooding attacks and BYE DoS attack authors of [163, 167] models the legitimate state transitions of SIP-based call setup and teardown process. Instead of collecting and deriving the call state and protocol dependent information from the packets, [167] utilizes the state machines of network protocols and the interaction among them for intrusion detection. The state transitions are triggered either by the arrival of packets or the internal interaction between protocol state machines.

A formal method based on Coloured Petri Nets [92] to design a SIP INVITE flooding attack detection framework is proposed in [116] and [43]. A distributed filtering architecture is designed in [88] to protect SIP devices against large scale flooding DoS attacks. A defense architecture against SIP Distributed-DoS floods based upon a redirection mechanism and a combination of source and destination traffic filtering is proposed in [179]. [3] proposes a real-time attack classification system using Naive Bayes and decision trees to detect application layer SIP flood attacks. A bloom filter based monitor for detection of flooding attack is proposed in [68].

2.3.2 Defense against Abuse of Service

Fraud in telephony service is the biggest example of abuse of service as it causes a substantial loss of annual revenue for many telecommunication companies throughout the world [18]. Fraud detection in the telecommunications industry requires the identification of a small fraction of fraudulent calls from the high volume of call traffic. This presents a significant research challenge in the design of efficient and effective fraud detection architecture. Diverse techniques are adopted by researchers for the prevention and early detection of fradulent activity in VoIP domain, such as signature-based [50, 157], rule-based systems [83, 85, 158], behavior profiling [157], Artificial Intelligence techniques like neural networks or decision trees [22,123,183], statistical methods [204], Gaussian mixture model [183], Bayesian networks [183], visual interface [34], machine learning [84] and data mining techniques [52]. A brief discussion of the fraud problem in VoIP networks and evaluation of the related available solutions is found in [72]. [148] presents a survey of the anti-fraud solutions proposed in various areas, and their usability in the VoIP context.

Some of the major fraud schemes and the techniques used to address them is discussed in [13].

Authors of [157] and [4] propose to obtain a *signature* of normal network behavior from historical calling activity. This *signature* helps to distinguish between normal and fraudulent usage of the VoIP services. Latent Dirichlet Allocation (LDA) is applied in [134,204] to build user profile signatures and assumes that any significant unexplainable deviations from the normal activity of an individual user is strongly correlated with fraudulent activity. The user activity is represented as a probability distribution over call features which surmises the user's calling behavior. This probability distribution is derived from LDA which can accurately describe user profiles by combining different classes of distributions. To score calls the authors compare the likelihood of the user generating a call versus a fraudster generating the same call.

A rule-based expert system is presented in [83] which aims to the detection of superimposed fraud cases in the telecommunications network of a large organization. Rules are induced by both using the network administrator's expert knowledge and by applying data mining methods on real world data.

An application of supervised and unsupervised learning approaches to telecommunications fraud detection is proposed in [84]. An unsupervised neural network approach [22] and data mining approach [52] are found to profile the behavior of mobile phone users for the use in fraud detection.

Authors of [183] performed experiments with three different methods to detect illegitimate usage of communication networks. Firstly, a feed-forward neural network based on supervised learning is used to learn a discriminative function to classify subscribers using summary statistics. Secondly, a Gaussian mixture model is used to model the probability density of subscribers' past behavior so that the probability of current behavior can be calculated to detect any abnormalities from the past behavior. Lastly, Bayesian networks are used to describe the statistics of a particular user and the statistics of different fraud scenarios.

2.3.3 Defense against Social Threat

Defense against social threat mainly refers to protection from SPIT (Spam over Internal Telephony). SPIT is considered the equivalent of email spam in VoIP environments delivered by means of voice calls. Thus, most of the approaches introduced so far for managing SPIT actually adopt and apply anti-email-spam techniques. Due to the real time communication constraint of SPIT calls, many effective email spam prevention methods are not effective at all in anti-SPIT system.

A survey of proposed anti-SPIT techniques proposed by researchers in recent years and evaluation of their effectiveness is found in [71, 118]. Rosenberg *et al.* [155] examines the various possible solutions that have been discussed for email and consider their applicability to spam call detection in SIP based VoIP services in an IETF draft RFC 5039.

Content Filtering

The most effective techniques of email spam protection content filtering where the content of messages is analyzed to characterize them as spam or not. Unfortunately, this type of spam filtering, while successful for email spam, is not effective for call spam due to the real time communication nature of VoIP communication. An analysis of the audio content is not appropriate since the call in question is already established when audio data are available. Furthermore, automatic methods based on speech recognition are currently too complex and language dependent to be deployed for VoIP calls.

This technique, however, can be used for off-line spam call detection such as detection of voice spam from subscribers voice mailbox proposed in [90]. For this purpose, the authors of [90] develop a speaker independent speech recognition system based on Dynamic Time Warping (DTW) [194] algorithm to make comparison of speech messages. A content analysis based SPIT detection and prevention method is proposed in [180]. Here, a robust audio fingerprint of spectral feature vectors is computed for incoming audio data. New calls are compared with these audio fingerprints to detect spam calls.

***Black*, *white* and *grey* lists**

Use of *black* and *white* lists is a simple and straight forward techniques for preventing email spam. Generally, the *black* list of a user contains the contact addresses of undesired callers. White lists are the opposite of black lists, it is a list of valid callers (e.g., family members, friends and collegues, etc) that a user is willing to accept call from. Unfortunately, the effectiveness of the *black* and *white* lists is limited for detection of SIP spam calls. *Malicious* users can easily pick a new caller ID to avoid the black list and *white* lists are susceptible to address spoofing though a strong identity authentication mechanism can prevent that problem. Use of a special list named *gray* list for SPIT detection is proposed in [171]. They proposed a Progressive Multi Grey-Levelling (PMG) mechanism that monitors call patterns and assigns a gray level to each caller based on the past call patterns for a specific window. If the gray level is higher than a certain threshold, the call is deemed SPIT and the caller blocked. Enhanced Progressive Multi Grey-Leveling (EPMG) is proposed in [17] for *gray* listing.

Payment at risk/Charging-based

SIP based VoIP services has become very attractive for spammer to transmit unsolicited calls as it allows lows cost telephony. Considering this, *payment at risk* technique is proposed by researchers where the idea is to affect the spammer financially. Forcing the spammers pay for each call will certainly reduce the volume of spam as the sending expenses will be higher and might exceed the achieved benefit. A possible scenario of *payment at risk* approach is described in [155], when a user X sends to user Y, user X deposits a small amount of money into users Y account. If user Y decides that the message is not spam, user Y refunds this money back to user X. If the message is spam, user Y keeps the money. A charged based spam call prevention framework is designed in [150]. However, building a charging model to support this technique and defines who collects the money and what are the payment policies is also a complex task. As a result.

Challenge-Response Based

By utilizing an automated procedure (i.e., bot) spammers can easily generate a huge amount of spam calls. A known way to protect SPIT from automated calling machines is *Challenge-Response* where the call is only established if the caller can prove that he/she is a human and not a bot. This is performed by *Turing test* [195] or Computational Puzzles where the caller is given a challenge, that a human can solve easily and that is hard to solve for a machine.

Hidden Turing tests is applied in [143] to received calls and compare them to typical human communication patterns (e.g., pauses between conversations, etc.) for distinguishing humans from automated calling machines. Authors of [104] proposed a technique using the text-based Turing test and demonstrated its compatibility with the SIP protocol. Authors of [176] evaluate existing audio CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) and suggest specific attributes-requirements that an audio CAPTCHA should meet in order to be effective. They demonstrate there is no existing implementation among the set of popular audio CAPTCHA suitable enough for VoIP environments.

To detect and prevent spam calls from automated machine a challenge-response concept is discussed in [91] where a SIP Proxy or User Agent Server can request from a User Agent Client (caller) to compute the solution to a puzzle. The idea is that the process of solving the puzzle will increase the CPU usage and thus will reduce the number of unsolicited calls that it can generate.

Reputation Based

This approach is based on the notions of reputation and trust between the callers and callee in the network. If the level of trust and reputation is above a pre-defined threshold then the communication is permitted, otherwise it is rejected. The trust and reputation level can be measured by past communications, information exchange between domains, and feedback from users. Authors of [74] suggested to build a *judgement model* to detect SPIT calls. The model uses the user feedback and integrates the trust with the reparation to judge a call. Here, trust indicates the authentic measurement from the callees and reputation indicates the authentic measurement from the non-callees. A trust mechanisms is proposed in [150] where each SIP user maintains a reputation list of his friends and a scores them according to the level of trust. A large social network is constructed by combining reputation lists of all the SIP users. This paper proposes that the reputation lists of the SIP users can be exchanged between SIP providers, though in reality exchange of information between VoIP providers may not possible due to user privacy agreement. The SPIT detection mechanism proposed in this paper is very simple, if a provider does not find any entry for a user from which a SIP message is received, it requests reputation information about this user from the other providers. If all replies are negative, then the received SIP message is considered as a spam. A collaborative reputation-based voice spam filter is proposed in [209] where reputation of a VoIP user is derived from his/her cumulative online duration.

Multistage Scoring Based

"VoIP SEAL" [160] filters SPIT calls by a two-stage decision process: The first stage has invisible (non-intrusive) modules that analyzes the information available before actually answering the call (e.g. the INVITE message), while, the second stage is the interacting (intrusive) that interacts with the caller and/or the callee. Each stage is composed form several modules, such as Turing tests, white/black list, simultaneous calls, call rate and URI's IP/domain correlation, user feedback, etc. Each module of the first stage contributes a score in $[-1, 1]$, where high score correspond to a high probability that the call is a SPIT. If the final score is higher than the lower threshold, then the call passes to the second stage modules. On the other hand, if the score is higher than the higher threshold, the call is rejected. The main feature of "VoIP SEAL" is the adoption of a modular architecture, and the easiness to add and update modules to respond to new or different kinds of SPIT. Two challenges of efficient deployment of "VoIP SEAL" is described in [15]: (i) the reliance on database queries makes "VoIP SEAL" hard to adapt it to an online, real-time, operation, and (ii) the need to store CDRs (Call Detail Records) for a specific analysis purpose poses supplementary deployment hurdles. The authors show that “VoIP SEAL” can be operated accurately by the stream-based VoIP-stream approach.

A modular simulator is presented in [51] which was built in order to test effectiveness of the SPIT prevention systems “VoIP SEAL” [160]. Authors of [122] concentrate specifically on unsolicited calls trying to isolate users that fall outside expected behaviors.

A multi-stages scoring framework is proposed in [106] which focuses on modeling of spam callers’ behavior to calculate the SPIT level for management of the quality of service. Authors of [38] proposed a Voice Spam Detector (VSD) combining many of the SPIT filtering techniques such as:(a) presence, (b) traffic pattern, (c) black and white lists, (d) Bayesian learning, and (e) social networks and reputation.

Social Behavior Profiling

The choice of receiving or rejecting a voice call depends on the social meaning of trust, reputation, friendship of the calling party and their own mood. That is why, in recent years, researchers are integrating the behavioral aspects of human to determine the legitimacy of voice calls.

Authors of [10], [26], [198] and [9] proposed to use duration of calls to establish social network linkages and global reputations for callers, based on which call recipients can decide whether the caller is legitimate or not. While, authors of [7,8] suggested that the computation of reputation should be two fold; firstly it should not involve user feedback and secondly it considers other network features in addition to call duration. They use the similarities and social ties among VoIP users for SPIT detection. The social ties between users is measured through features like out-degree, number of repetitive calls, reciprocity and interaction rate. A multi-stage, adaptive spam filter based on presence (location, mood, time), trust, and reputation is described in [108] to detect spam in voice calls.

Call Pattern Profilng

SPIT detection methods using call pattern analysis is proposed in [110,119]. Authors of [110] express call patterns with parameters such as user relationships, average call duration, and the rate of unsuccessful calls. Authors of [165, 166] presented two approaches for SPIT call detection based on the distributions of selected call features (i.e., day and time of calling, call durations etc.)

Knowledge-based Approach

Firewalls to detect SPIT for VoIP devices is designed in [205] and [117] that uses attack signature to identify call invitations from suspicious user agents. Authors of [126] addressed the SPIT detection problem using decision tree. Conceptual models is proposed in [44, 70] to formally described the SPIT domain. The formal representations included capabilities, such as modeling the SPIT phenomenon in a SIP-based VoIP environment, a common understanding of SPIT domain, as well as reusable SPIT-related knowledge interoperability, aggregation and reasoning. A formal verification method for validating the effectiveness of the anti-SPIT system is found in [175]. A protection system to enable personalized and role-based SPIT prevention is proposed in [40].

Machine Learning & Data Mining Techniques

Machine learning and data mining techniques [3, 188, 203] are successfully used for classification of SPIT calls. Authors of [188] proposed a multi-feature (e.g. call frequency, average call duration, etc) call pattern analysis with unsupervised Random Forests classifier. Wu *et al.* [203] proposed detection of SPIT calls through clustering based on the call parameters, using optional user feedback for some calls. A semi-supervised learning algorithm is used here for this clustering process which also considers unlabeled data. In the context of VoIP, semi-supervised learning algorithms is very effective as this is free from the laborious and error-prone manual parameter configuration.

Miscellaneous

Authors of [136] focused on reducing the false negative in SPIT call identification. The authors discussed that most legitimate calls a person received are from persons or organizations with strong social ties such as friends, family, colleagues, etc. Some legitimate calls, however, are from those with weak social ties such as a restaurant the callee booked a table on-line. Since a callee's contact list usually contains only the addresses of persons or organizations with strong social ties, filtering out unsolicited calls using the contact list is prone to false positives. After analyzing call logs they identified that legitimate calls are initiated from persons or organizations with weak social ties through transactions over the web or email exchanges. Considering this, they proposed a special mechanism to detect legitimate calls from person with weak social ties using cross-media relations to prior contact. In this mechanism, a potential caller offers the callee his contact addresses which might be used in future calls. On the other hand, a callee provides a potential caller with weakly-secret information that the caller can use in future calls in order to be identified as someone the callee has had prior contact through other means.

Authors of [95] also focused on the performance of a SPIT detection system. Particularly they concentrated on ways to design SPIT filters with the two features: (i) performance guarantee (e.g., percentage of mistakes of the filter in the worst case), and (ii) reduction of manual configuration.

2.4 Discussion

Due to the different characteristics of attacks a large body of research has been dedicated to attack-centric solutions. Most of the cases, solutions designed for detecting one specific attack are not efficiently applicable in detecting other attacks. For instance, SPIT detection techniques are not fully suitable for detecting flooding DoS attacks. Thus, the VoIP domain is in need of a unified platform that is able to detect and protect SIP services from different attacks.

Threshold based approaches are widely used in recent literature for detection of attacks. An adaptive threshold based approach considering the dynamic nature of traffic flow is very effective is detection of attacks such as DoS, SPIT, fraud, etc. However, most of these techniques suffer from setting a suitable threshold to distinguish between the legitimate and anomalous traffic. This makes good performance of these threshold based solutions in the real world questionable by increasing the possibility of high false negative rate. Moreover, threshold based approach does not always work very well since a careful hacker can evade this security measure by crafting an attack that keeps the CPU usage below the threshold, while still interfering with the genuine network traffic.

Signature/rule based technique is another popular approach adopted by researchers for intrusion detection. This technique provides the opportunity to easily detect different types of attacks. However, handling of intrusion/anomalous detection problem in the context of SIP with a table-drive approach (rule/signature database) is destined to run up against the combinatorial explosion, as there are endless ways of forming a malformed or malicious message. Again, the stochastic nature of arrival rate of call requests that varies with time cannot be easily modeled by a deterministic process with a time-varying rate and thus developing an efficient attack signature remains a challenge. Formal techniques, such as state machines based on the network protocols and the interaction among them are also utilized for intrusion detection. This approaches, however, are memory intensive as many state machines are required for protection against various attacks.

Machine learning and data mining techniques seem promising for traffic analysis and network intrusion detection, particularly, in the detection and prevention of fradulant traffic and SPIT calls. However, effective deployment of machine learning and data mining based intrusion detection depends on a deep semantic insight into a system's capabilities and limitations.

Though a number of security control systems for the attack detection and protection of SIP based services are available in recent literature, lack of sufficient dataset (e.g., VoIP traffic) affects the proper performance testing of these systems and also limits the possibility of comparing the results of different security control systems.

These considerations mandate two different, yet related, lines of research and development. First of all, filtering architectures able to analyze the syntax of protocols and correlate it with the potential semantic of the service are needed to control the traffic. These tools will be initially developed based on heuristics considerations, i.e., analysis modules will be developed based on rule-of-thumb considerations on users' and protocols' behavior, since presently there is no theoretical framework that can drive their development. Second, synthetic models for generating traffic are needed to match real world traffic characteristics, to build a solid theoretical insight on the system behavior, and finally to cross-check and tune the heuristics analysis tools.

Chapter 3

Building a SIP Analysis Methodologies

3.1 Formal Representation of VoIP Security Domain

The pre-requisite of designing any security measure for VoIP system is to obtain some knowledge about the security challenges in the VoIP domain. A single taxonomy is not likely to be definitive, thus, various categorizations of potential attacks against VoIP services are defined by researchers using several different viewpoints and mapping the vulnerability space along several axis. In order to enable the domain experts to share, utilize and exchange this knowledge, it is necessary to have these attack taxonomies represented in some formal way followed by specific security recommendations and guidelines for protecting the underlying infrastructure from these attacks.

Ontologies [53, 73] are widely used in different fields (e.g., artificial intelligence, the Semantic Web, systems engineering, software engineering, etc) to formally represent knowledge as a set of concepts within a domain, using a shared vocabulary to denote the types, properties and inter-relationships of those concepts. *Ontological* approach is popular for knowledge representation due to its numerous advantages over taxonomies and other classification schemes. Through an ontology a common information structures of a domain can be formed, knowledge can be reused, assumptions within a domain can be made, and various semantic relationship among different concept of the domain can be explored.

In recent years, *ontologies* seem extremely promising in knowledge representation also in the field of information and communication security [14, 54, 82, 96, 146, 196]. These *ontologies* can be also employed in a real environment for intrusion detection, security testing and vulnerabilities identification purpose [12, 114, 147, 189, 190]. Specially, the use of an *ontology* seems promising in intrusion detection in distributed environments [2] where the system and its constituent components should have a common mechanism to share the collected knowledge about security attacks and their countermeasures.

[44, 67] present that ontology-based security policies can be also applied in a real VoIP environment. Authors of [44] demonstrate that the proposed ontology, combined with a set of attack identification criteria, as its underlying axioms and rules, could support intrusion detection process of the SIP based VoIP services.

Inspired by above research, we introduce an *ontology* named “VoIP-Onto”, a common vocabulary to exchange security related information of the VoIP domain. “VoIP-Onto” is an attempt to define a comprehensive taxonomy of potential attacks in a formal way by combining different taxonomies proposed in related literature. It models the categorizations of potential attacks, threats and vulnerabilities against VoIP services, countermeasures and their relations. The ontology can be used as a general vocabulary, road map, and extensible dictionary of the domain of information security. Chapter 4 describes the content of the “VoIP-Onto” as well as its usages, potential for extension, technical implementation and tools for working with it.

3.2 SIP Message Filtering Methodology

The first step of our SIP analysis and anomaly detection process is the control of single messages to determine if they are "good" or "bad" messages so that the latter can be discarded, thus avoiding the possibility that they can cause the execution of harmful procedures. The distinction between "good" and "bad" SIP messages is a somewhat fuzzy concept, since there are many different ways for a message to be "bad". Fig. 3.1 shows a simple classification tree to clear the terminology of "good" and "bad" SIP messages that we have used in this thesis.

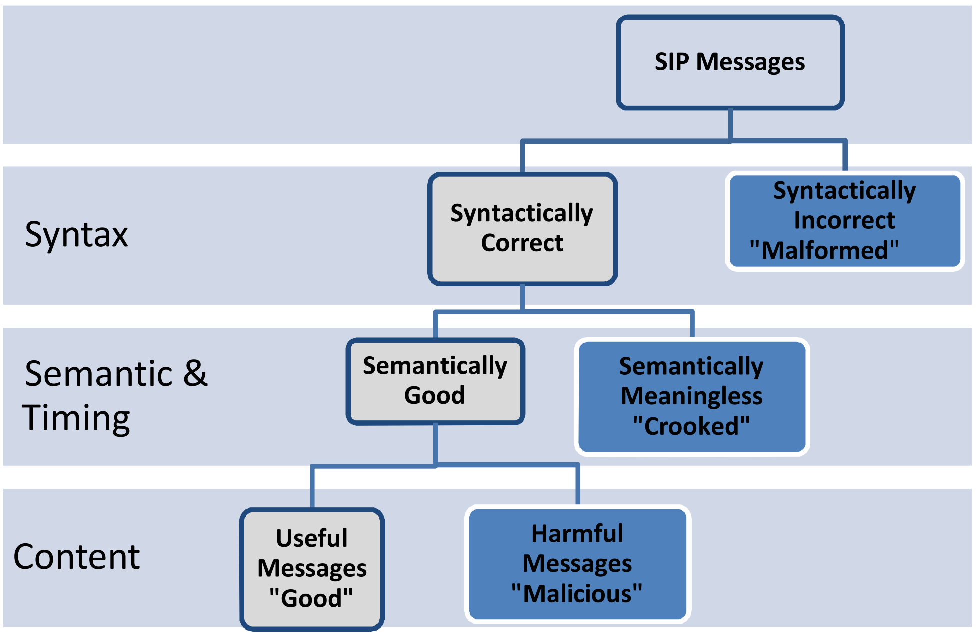


Figure 3.1: Simple binary classification of SIP messages highlighting the terminology used in the thesis

A "good" message is simply a valid SIP message that can be correctly interpreted by its recipient. This means the message is syntactically correct, semantically meaningful, and comes at the right time to trigger a correct and useful application decision. From a theoretical point of view, "bad" messages are the complement of the set of "good" messages, but this will not help much in the classification process. We define the set of "bad" messages as the union of "malformed", "crooked", and "malicious" messages. "Malformed" messages are those that simply are syntactically wrong. "Crooked" messages are those that, while syntactically correct, have no meaning, cannot be interpreted, are ambiguous, or lead to a dead-lock, etc. Finally, "malicious" messages, are those that are syntactically correct and semantically meaningful, but will harm the system: normally these are forged on purpose, but they can also be the outcome of mal-functioning devices, badly implemented instances of the protocol or, more likely, of its extensions.

Because of the different nature of the structure, contents and timing of these "bad" messages, their classification is best carried out by specialized detectors: a multistage classifier. Though our proposed ontology "VoIP-Onto" can be employed in a real environment for testing or intrusion detection purposes, it is a tough job to define a comprehensive set of rules in order to detect the diverse and endless form of "bad" messages. Considering this, we limit the scope of our proposed ontology only for sharing domain knowledge about the taxonomy of VoIP attacks and guidelines of countermeasures in the hope that it will serve as a starting point for the new researchers to understand security risks in the VoIP domain.

The multistage architecture allows better separation of the intrinsically different tasks and offers the possibility of pipelined parallelism on separate hardware if message traffic volumes warrant or require classification rates that cannot be achieved by a single processor. Fig. 3.2 shows the logical architecture of the SIP traffic analysis & filtering system. The proposed architecture is a lightweight application and can be installed on the same machine of the agent, or, if performance requires it, as a kind of firewall in front of it.

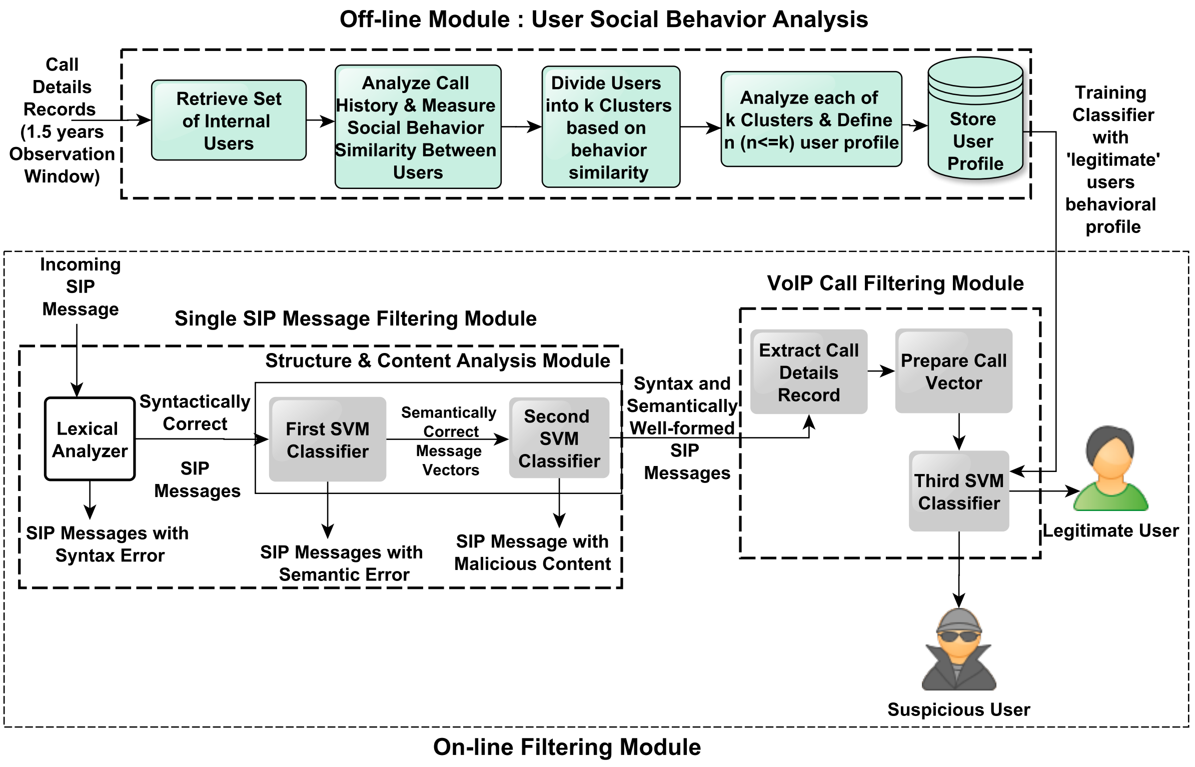

Off-line Module : User Social Behavior Analysis
Call Details Records (1.5 years Observation Window)
Retrieve Set of Internal Users
Analyze Call History & Measure Social Behavior Similarity Between Users
Divide Users into k Clusters based on behavior similarity
Analyze each of k Clusters & Define n (n<=k) user profile
Store User Profile
Training Classifier with 'legitimate' users behavioral profile
Incoming SIP Message
Single SIP Message Filtering Module
Structure & Content Analysis Module
Lexical Analyzer
Syntactically Correct SIP Messages
First SVM Classifier
Semantically Correct Message Vectors
Second SVM Classifier
SIP Messages with Syntax Error
SIP Messages with Semantic Error
SIP Message with Malicious Content
Syntax and Semantically Well-formed SIP Messages
VoIP Call Filtering Module
Extract Call Details Record
Prepare Call Vector
Third SVM Classifier
Legitimate User
Suspicious User
On-line Filtering Module

Figure 3.2: System Architecture for SIP message classification

3.2.1 Lexical Analyzer

The first stage filtering is performed by the lexical analyzer which investigates each SIP message to determine if they are part of the language generated by the formal grammar which specifies the SIP protocol [156]. This straightforward tool follows a deterministic and efficient process to identify and discard all mistakes and malformations that violate the grammar.

3.2.2 Support Vector Machine for Classification

SIP messages that have passed the lexical analyzer filter may still be "bad" as they can be semantically meaningless or can carry harmful content. Detection of these "bad" messages is a more complex task and requires a more delicate handling, as it is not a sharp decision whether a message is semantically meaningful or not. Any hope of tackling this problem with an algorithmic or table-drive approach is destined to run up against the combinatorial explosion of the cases that need to be considered, as there are endless ways of forming a "crooked" or "malicious" message. We need to consider that there are 14 request message types, 6 response message kinds (discussed in sub-section 1.2.3), there are no upper limits on some values such as the length of headers, that some fields are mandatory while other are optional and finally there are multiple ways of structuring a correct SIP message. This results in the practical impossibility of systematically examining all possible cases of corruption casual or voluntary of a message.

For these reasons, we turned our attention to machine-learning techniques in which an automaton can "trained" by being shown a sufficiently rich set of "good"/"bad" examples. After the automaton has been adequately trained (for instance, until the classification error percentage falls below a given threshold), it can be used to classify messages that were never seen before. An added bonus of this approach is in its flexibility to fit new kinds of "bad" messages that might become common at a later time, perhaps as a new breed of malicious messages is introduced when some weakness of the protocol is uncovered. The adaptation to the changed operating scenario can be obtained simply by retraining the machine automaton while including the new messages, duly identified as bad.

Various approaches to supervised machine learning have been proposed. Recently the so called Artificial Neural Networks (ANN) and the Support Vector Machines (SVM) have received much attention [105,125] since they have performed quite well in a variety of problem contexts. For our second and third stage filter, we have selected to use a SVM for the reasons that will be discussed below. SVMs have been introduced by Vapnik in [192] and have been successfully applied to many fields such as Bio-informatics, Natural Language Processing, Handwritten Character Recognition and many others. Unlike Artificial Neural Networks, SVMs do not have the problem of getting stuck in a local minimum while searching for an optimal configuration of its operating parameters (see for example [23, 25, 170]). In addition, SVMs are scalable since the computational complexity does not depend on the dimensionality of the input space. Finally, SVMs optimization by training (or re-training) is quite fast, which can represent an important point if online intrusion (re)configuration is of paramount importance.

3.2.3 Support Vector Machine Basics

The basic idea of SVM classification is to interpret the d-dimensional feature vectors derived from SIP messages as points in an d-dimensional space. Some of these points correspond to “good” messages (label them as -1) and the others correspond to “bad” messages (label them as $+1$). The classification problem can be seen as finding an hyper plane that separates the space in two sub-spaces: one containing all the -1 points, the other all the $+1$ points. If the set of points is linearly separable into two classes, there are infinite planes that will work. However, there is only one “best” hyper plane that maximizes the distance between it and the nearest data points of each class. Unfortunately, it is often the case that no such hyper plane exists (the set of points is not linearly separable) and hence some points would be misclassified, as they would lay on the “wrong” side of the best hyper plane. However, while not linearly separable, the points could be separable if some other, more complex, surface were used instead of the (simple) hyper plane. Informally, the SVM computation does this by projecting all points into a higher-dimensional space and in that space the complex separating surface becomes a hyper plane, thus linearly separating the set of projected points.

More precisely, given a set of n SIP messages, let $\vec{x_i}$ be the i-th message which is transferred into a d dimensional feature vector and let $y_i \in \{-1, +1\}$ be and indicator function where -1 indicates that the i-th message belongs to the class of "good" message class while $+1$ indicates that the i-th message is in the "bad" message class. The equation of the hyperplane separating the training set $\left\{(x_i, y_i)\,|x_i \in \Re^d, y_i \in \{-1, 1\}\right\}_n^{i=1}$ can be defined as:

$$\vec{w} \cdot \vec{x} + b = 0 \tag{3.1}$$

where $\vec{w}$ is the vector normal to the hyperplane and $\frac{b}{\|\vec{w}\|}$ is the perpendicular distance from the hyperplane to the origin.

For the linearly separable data, SVM finds the optimum separating hyperplane with the largest margin (Fig. 3.3) by solving the Quadratic Programming (QP) optimization problem described by eq. (3.2).

$$min\left\{\frac{\|\vec{w}\|^2}{2}\right\}, \text{ subject to } y_i\left(\vec{w} \cdot \vec{x_i} + b\right) \geq 1 \ , \forall_i \tag{3.2}$$

If the set of points is *not* linearly separable, then instead of trying to fit a non-linear model, the set of points can be mapped to a higher-dimensional space by a non-linear mapping function ϕ, $\vec{x} \to \phi(\vec{x})$ so that the points become linearly separable in this higher dimensional space. The classification function in dual space becomes:

$$\begin{aligned} h\left(x\right) &= sgn\left(\phi\left(\vec{w}\right) \cdot \phi\left(\vec{x}\right) + b\right) && (3.3)\\ &= sgn\left(\sum_{n}^{i=1} \alpha_i y_i \phi\left(\vec{x_i}\right) \cdot \phi\left(\vec{x}\right) + b\right) && (3.4)\end{aligned}$$

In the quadratic optimization problem for non-linearly separable data SVM, the training vectors appear only in the form of dot products, $\left(\phi\left(\vec{x_i}\right), \phi\left(\vec{x_j}\right)\right)$ which imply that computationally expensive dot product calculation is needed. However, by using the so called "kernel functions" one can apply a "kernel trick" that avoids the expensive dot products. Kernel functions that have been favored in the recent literature include:

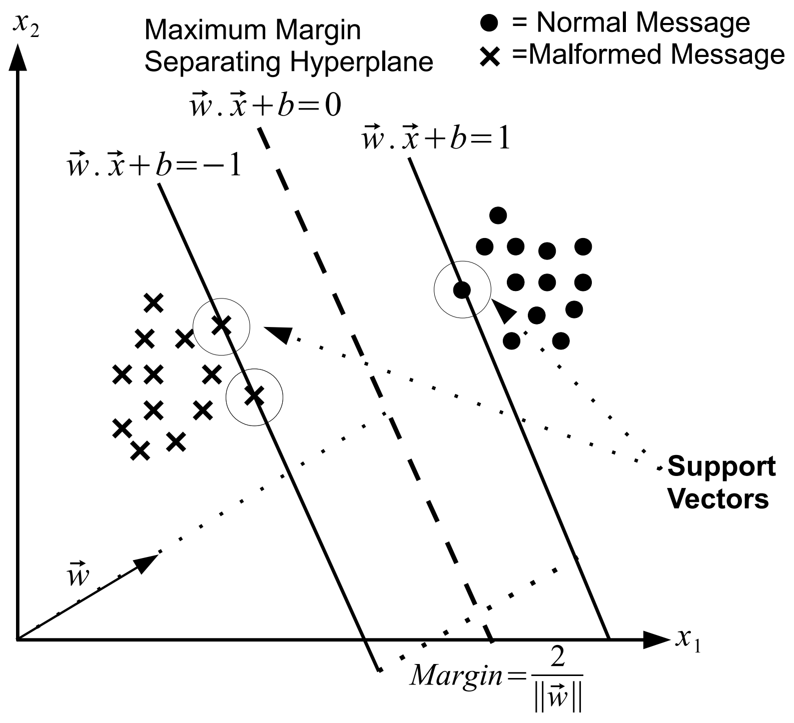


Figure 3.3: Linear Support Vector Machine: optimum separation hyperplane

- *Linear kernels*: $k\left(\vec{x_i},\vec{x}\right)=\vec{x_i}\cdot\vec{x}$;
- *Polynomial kernels* of degree d: $k\left(\vec{x_i},\vec{x}\right)=\left(\vec{x_i}\cdot\vec{x}\right)^d$;
- *Radial Basis Functions (RBF) kernels*:
  $k\left(\vec{x_i},\vec{x}\right)=exp\left(-||\vec{x_i}-\vec{x}||^2/2\sigma^2\right)$.

Thus, the classification function of eq. (3.3) becomes:

$$h\left(x\right) \;=\; sgn\left(\sum_{n}^{i=1}\alpha_i y_i k\left(\vec{x_i}\cdot\vec{x}\right)+b\right) \tag{3.5}$$

Fig. 3.4 represents the polynomial projection of non-linear data into high-dimentional feature space where they are linearly separable.

3.2.4 SIP Traffic Analysis and modeling

While the identification of syntactically incorrect SIP messages is straightforward (e.g., either a messages belongs to the language defined by the

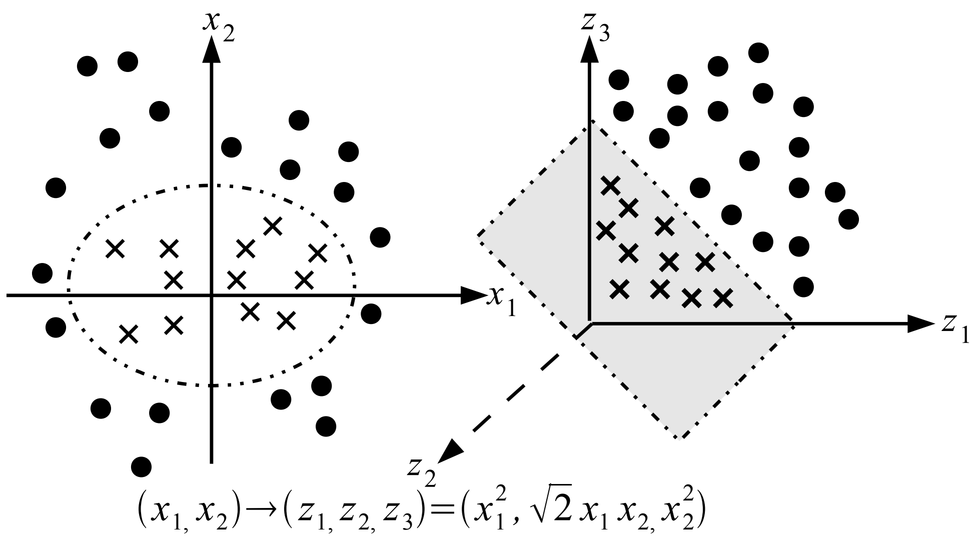

x_2
x_1
z_3
z_1
z_2
$(x_1, x_2) \rightarrow (z_1, z_2, z_3) = (x_1^2, \sqrt{2} x_1 x_2, x_2^2)$

Figure 3.4: Non-linear SVM: polynomial mapping

protocol [156] or it does not), detection of semantically meaningless and harmful content requires the classifier to be integrated and trained with the information about the legitimate system and users behavior. It requires to correlate different messages to gain a thorough knowledge about the system behavior. In fact, without a deep knowledge of the normal behavior of the network and users only major service failures would be detectable, and even these would still require huge amount of time for their root cause identification. Even worse, silent problems (e.g., telemarketers sending unsolicited communications using bot machines, billing fraud, a malicious user attempting to impersonate another user during a call, etc) not leading to an immediate service meltdown would be completely untraceable reducing the users' confidence and satisfaction in the overall service. For instance, detection of unsolicited calls by checking the single message without having any prior knowledge about the normal behavior of the system and users is not possible. It requires information about the social interaction pattern of both the caller and the receiver, and also the trust relationship between them to suspect a call as spam before attending. In VoIP domain, the legitimate VoIP callers usually have specific calling patterns toward their friends and family members, while, a telemarketer or spam caller usually collects callees identities by crawling the web or reading telephony directories and normally results in a non-connected social network. The social interaction patterns of these callers are different from those of legitimate callers. This difference in social behavior of normal and spam callers reveals interesting information useful for identifying spam callers.

To this end, we have focused on an in-depth analysis of the network to learn the normal behavior of the system. We use the Social Network Analysis techniques in the stream of SIP messages to exploit different levels of behavioral patterns of users. Details about the social behavior analysis and modeling is presented in chapter 7.

3.2.5 SVM Classifiers

The second and third stages of our multistage classifier are meant to detect those "crooked" and "malicious" messages. Here, two separate SVM classifiers are used due to differences in nature and detection process of "crooked" and "malicious" messages. Furthermore, multistage SVMs have been considered more effective than single stage SVM in other classification tasks (see for example [115] [99]).

The SVM classifiers are trained with the normal behavior of the system and users observed from traffic analysis. The machines, thus trained, are capable of identifying semantically meaningless and malicious content and users. Details about the configuration of the SVM classifiers and the detection accuracy is described in chapter 6 and 7.

Chapter 4

Ontology of VoIP Security Domain

As mentioned in section 3.1, ontologies[1] are suitable for formalization of information about VoIP threats and also modeling security control systems. Ontologies rely on well-defined and semantically powerful artificial intelligence concepts such as description logics and provide explicit formal specifications of the terms in domains and relations among them [73], and an extremely promising paradigm in computer security field.

Though a small body of research has been dedicated to analyze the use of ontology in modeling network and computer attacks, very few attempts are found [66, 67] focusing on the formation of ontology of VoIP domain. [66] introduces a formalization which can be used for the identification of illegitimate behavior of SIP-based VoIP services. The authors only consider dos (denial of service) attacks rather than providing a general structure of possible attacks against VoIP system and their detection mechanisms.

To counteract this lack, we define an ontology named "VoIP-Onto" which can be considered as a conceptual model of VoIP threat domain considering the semantic relationship among security threats, attacks and system environment. Defining such an ontology is challenging especially when syntax and semantics are not sufficient to describe legitimate and anomalous behavior in VoIP networks. The following sections describe the

[1]Ontology, `http://semanticweb.org/wiki/Ontology`

content and scope of the “VoIP-Onto” as well as its usages in VoIP security.

4.1 “VoIP-Onto”:An ontology of VoIP domain

“VoIP-Onto” is an ontology of VoIP domain with an emphasis on the security perspective.

4.1.1 Domain and Scope of “VoIP-Onto”

The domain of “VoIP-Onto” includes the definition of both the legitimate and anomalous behavior of VoIP systems. It focuses on defining a comprehensive VoIP attack taxonomy including the possible ways to detect the occurrence of those attacks in VoIP systems.

The scope of this ontology is not only limited to sharing and inferring security relation information of VoIP domain but also can be extended for security testing, vulnerabilities identification and also anomaly/intrusion detection purposes.

4.2 Structure of “VoIP-Onto”

“VoIP-Onto” presents a structured way to view attacks against VoIP systems by subsuming a strong taxonomy of attacks. In constructing the ontology “VoIP-Onto”, we consider the VoIP threat taxonomy described in 2.1 where the threats against VoIP are divided into four categories: (i) interruption of service threats, (ii) social threats, (iii) interception and modification of traffic threats, and (iv) abuse of service threats. Fig. 4.1 shows the taxonomy of attacks that is followed in “VoIP-Onto”. Another focus of this ontology is to include all the important information (target, source, mode, vulnerability, affected security criteria, etc) about the attacks and to put emphasis defining important features for detecting these

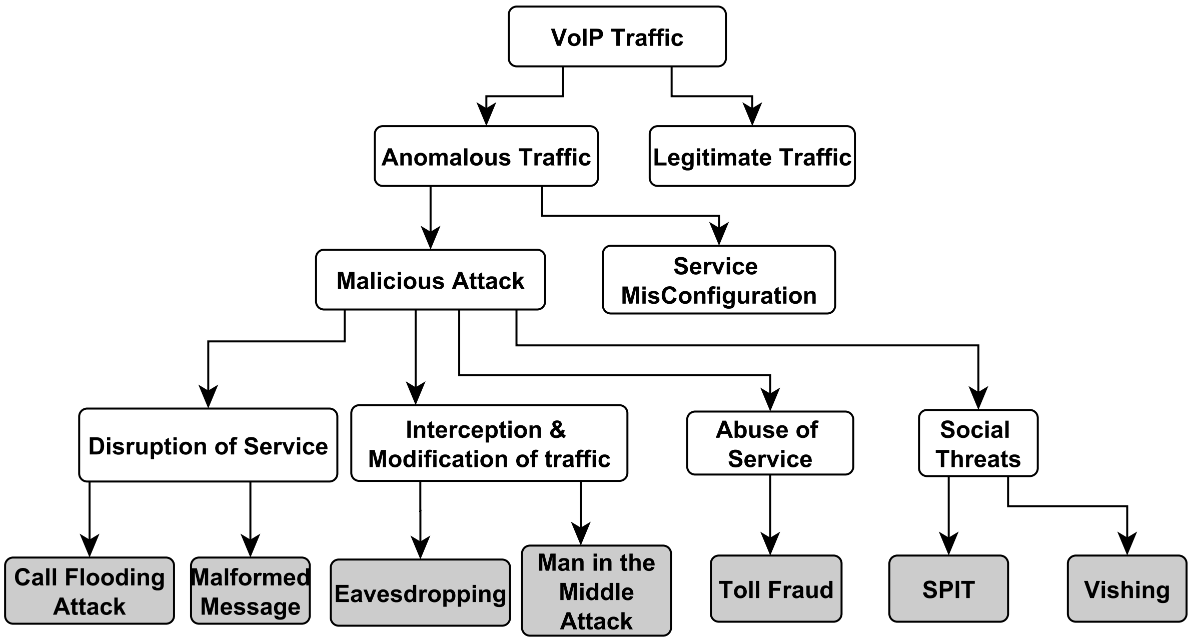


Figure 4.1: Taxonomy of attacks against VoIP domain subsumed by “VoIP-Onto”

attacks.

4.2.1 Concepts and Roles in “VoIP-Onto”

In “VoIP-Onto”, attacks against VoIP domain and various information about attacks (e.g., source, target, vulnerabilities, motivation, affected security criteria, etc) are considered the *concepts*, and the semantic relation between these concepts are the *roles* that are defined in the ontology. Fig. 4.2 shows the Entity-Relationship (ER) diagram of VoIP threat model where the semantic relation between the concepts are defined. This ER diagram can be considered as the high-level view of the proposed ontology. The threat model in Fig. 4.2 defines the legitimate behavior in the system as *Legitimate traffic* while any deviation from legal behavior is identified as *Anomalous Traffic*. Attacks against VoIP systems are the main source of anomalous behavior though abnormal traffic can also be transmitted in the

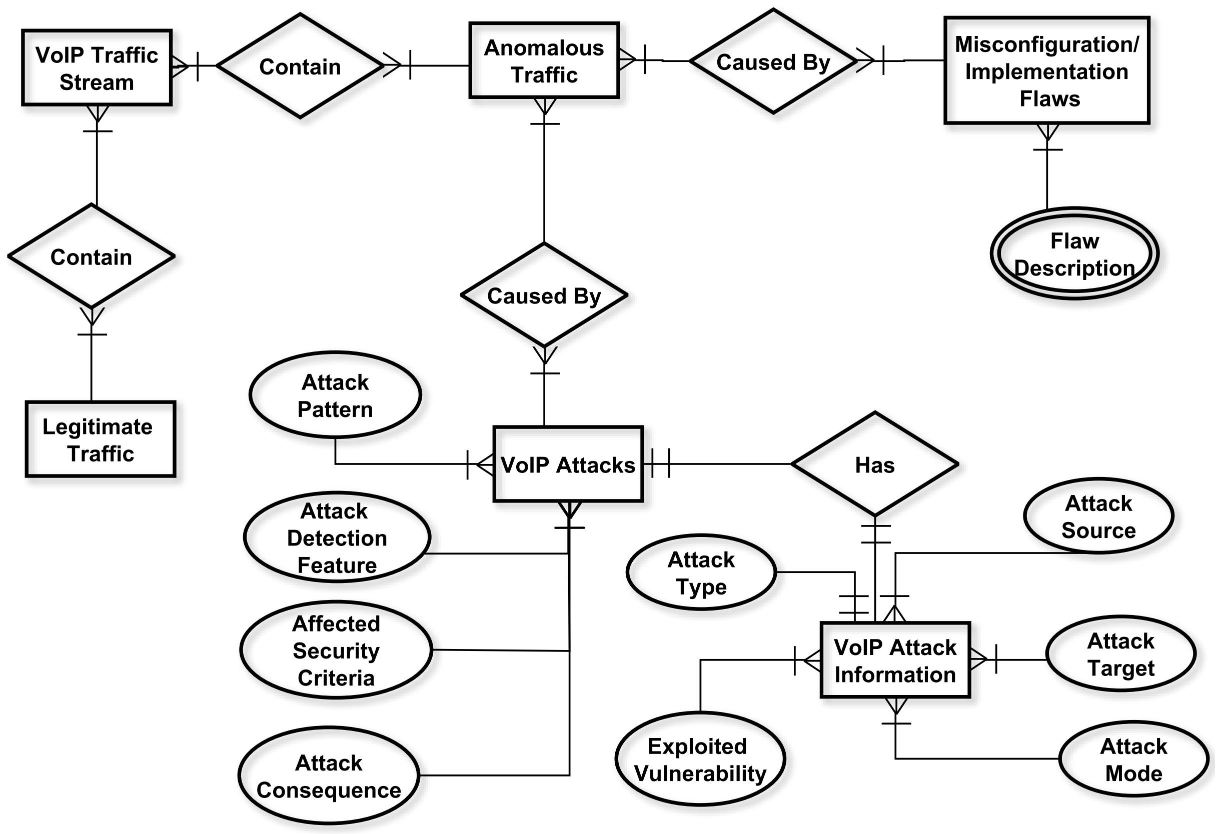


Figure 4.2: Entitiy-Relation diagram of VoIP domain

network due to implementation flaws or misconfiguration of the system. In order to determine the general structure of possible attacks against VoIP system, we consider several different attributes of attacks such as target, location, exploited vulnerabilities, etc and also reveal the semantic relation between these attributes of attacks. "VoIP-Onto" uses well known terms of VoIP systems as concepts and to define axioms. The domain of "VoIP-Onto" mainly includes the concept of *VoIP_Traffic* which indicates the overall traffic of a VoIP system. The class/concept *VoIP_Traffic* stream can be considered as the union of *Legitimate_Traffic* and *Anomalous_Traffic*.

- **Legitimate_Traffic**–The concept *Legitimate_Traffic* indicates the class of traffic that represents the normal legitimate flow of the newtork.

- **Anomalous_Traffic**–The concept *Anomalous_Traffic* refers to complement of the class of *Legitimate_Traffic*. This class contains traffic that is different from the legitimate normal flow of the network. Traffic stream of this class may be caused by various VoIP attacks or abnormal situation. So the concept *Anomalous_Traffic* can be defined as the union of *VoIP_Attack_Traffic* and *Abnormal_Traffic*.

The concept *VoIP_Attack_Traffic* indicates the anomalous behavior of the system due to attacks against VoIP system. Significant information about VoIP attacks and rules of detection and prevention them are used to define the concept of *VoIP_Attack_Traffic*. *VoIP_Attack_Info* is the concept/class that is used to represent important attributes of VoIP attacks. “VoIP-Onto” is modeled considering the network administrator’s view point and thus special attention is given to information about attacks such as effect of the attacks, affected security criteria, and rules for detecting attacks. Attributes that are necessary to model VoIP attacks are described below:

- **Attack Motive**:This attribute represents the motivation of attackers for launching an attack. Usually, the underlying motives of attackers to attack VoIP application is to degrade the service, disturb users with transmission of unsolicited messages, to steal confidential and private information, to be benefited financially, etc.

- **Attack Impact/Effect**:This attribute indicates the after-effect of the attacks. For example, attacks of the group *interruption of service threats* usually they try to degrade/stop the service quality by bringing it down.

- **Exploited Vulnerability**:There exists a close relation between threat and vulnerability. Threats can be considered as the potential violations of security while vulnerability is defined as a flaw or weakness in a system's design, implementation, or operation and management that could be exploited to violate the system's security policy. For a threat to be effective, an associated vulnerability must exist that ultimately can be exploited. If the vulnerability does not exist or it is not possible to be exploited, the threat is categorized as minimal. So we consider information about vulnerability as an important attribute of VoIP attacks and consider the relationship between vulnerability and threats in the ontology. According to [101], attacks can be occurred exploiting the following vulnerabilities in VoIP system:
  - **Flaws/Weakness_in_Protocol**–This concept/category describes vulnerabilities which are caused due to flaws or weakness in protocol design.
  - **Flaws_in Network Desing & Implementation**–Different flaws in network design and implementation is considered in this group.
  - **Poor_or_Misconfiguration_of_System**–Configuration errors of the services is considered in this group.
- **Attack Source**:This attribute informs about the number of the attackers. Attacks can be occurred by single attacker or there can be distributed sources to perform attacks. In the ontology, both single and distributed source of attacks are considered.
- **Attack Target**:This attribute reveals the target of attacks. Attacks can affect end users or servers.
- **Affected Security Components**:This attribute reflects the affected security criteria by the attacks. We consider widely used security

benchmark which is CIA (confidentiality, integrity and availability).

- **Attack Mode**:This attributes indicates the mode of operation of attackers. *Active* mode indicates that attacks are executed directly, while *Passive* mode refers to inactive mode of operation.
- **Attack Prevention Mechanism**:This attribute refers to the most effective mechanisms adopted by the researchers to detect and prevent a VoIP attack.
- **Features for attack detection**:This attribute mentions few significant features for detection of an attack.

Fig. 4.3 shows the semantic relation of VoIP attack information in "VoIP-Onto".

4.3 Used Tool for "VoIP-Onto" Implementation

"VoIP-Onto" is implemented using Prot*é*g*é*[2], an open source semantic editor for ontology development. Particularly, The Prot*é*g*é*-OWL editor[3], an extension of Prot*é*g*é* that supports the Web Ontology Language (OWL) which is a standard ontology languages endorsed by the World Wide Web Consortium (W3C)[4], is used here. For reasoning the ontology, Pellet[5], a semantic reasoners, is used.

4.4 Usage of "VoIP-Onto"

This section demonstrates possible usages of proposed ontology "VoIP-Onto". This ontology is helpful for the network administrator in identifying

[2]Prot*é*g*é*, http://protege.stanford.edu/
[3]Prot*é*g*é*-OWL editor, http://protege.stanford.edu/overview/protege-owl.html
[4]World Wide Web Consortium (W3C), http://www.w3.org//
[5]Pellet, http://protege.stanford.edu/

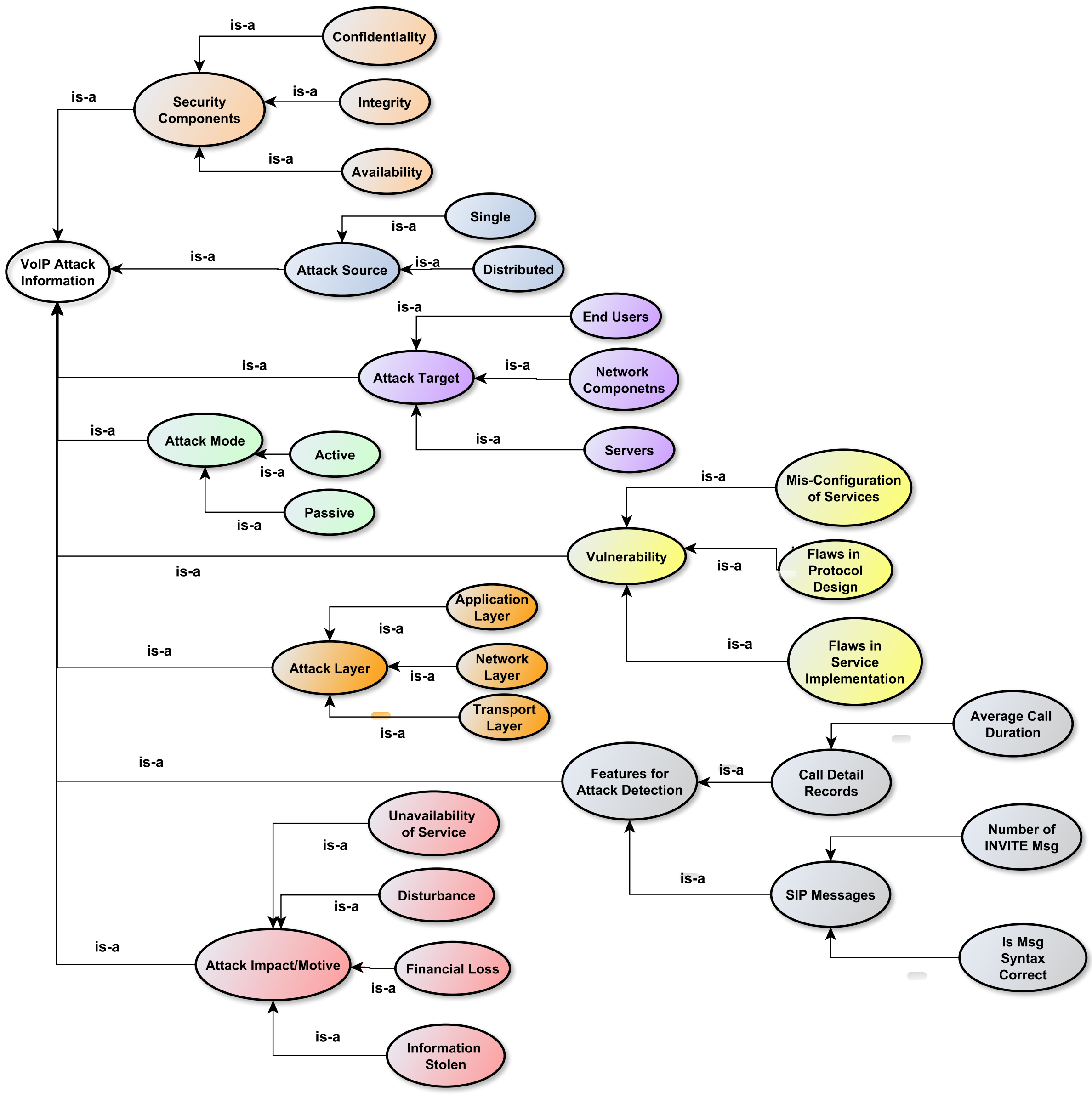


Figure 4.3: Attributes of attacks for defining VoIP attack domain.

the general pattern of vulnerabilities and attacks of VoIP systems. As a result it can be extended and employed in a real environment for network security testing or intrusion-detection purposes.

4.4.1 Sharing of knowledge

"VoIP-Onto" consisting of a vocabulary used to describe security threats of VoIP domain, an explicit specification of the intended meaning of the vocabulary and also a specific taxonomy clarifying how concepts can be used for capturing additional knowledge about the domain. Knowledge about VoIP specific attacks can be retrieved from the ontology by inferring and querying. Examples of query and reasoning the ontology for retrieving information about attacks are shows below.

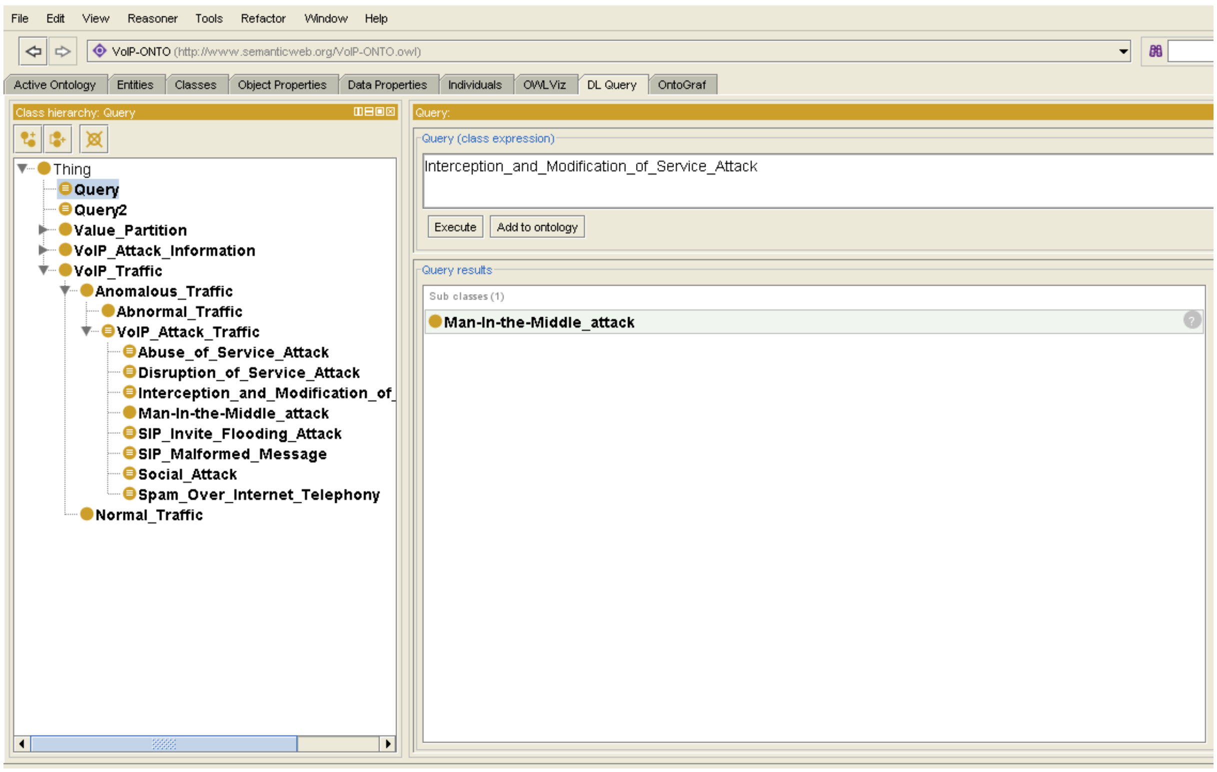
- Query 1:**Find an example of Interception and Modification attack**
  Response 1:**Man in the Middle** Attack

- Query 2:**Find Two possible features for detecting INVITE Flooding attacks**
  Response 2:**(i)Check Message syntax, (ii) Check number of INVITE Requests**

4.4.2 Ontology for Designing Control Systems

Use of "VoIP-Onto" can be extended by incorporating this ontology in security control systems such as intrusion detection systems. This can be done by considering the ontology as a rule database to support the intrusion detection process of the VoIP services. In this context, the attack identification criteria defined in "VoIP-Onto" is considered as the underlying axioms and rules of the rule based-intrusion detection systems.

The advantage of using the ontology as a rule database for rule based-intrusion detection is that before deployment it is possible to formally verify the consistency of the rules and axioms with the help of a semantic reasoner. This formal verification (e.g., to check the consistency, satisfiability



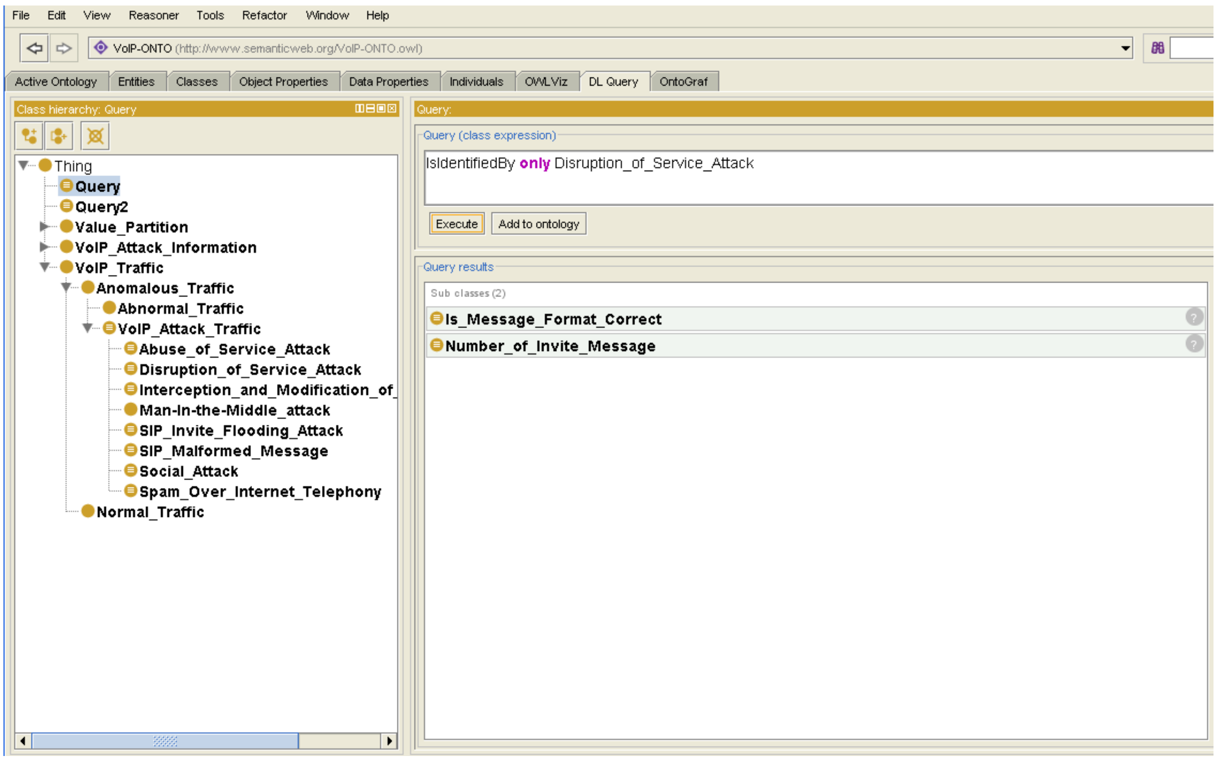
```
Thing
    Query
    Query2
    Value_Partition
    VoIP_Attack_Information
    VoIP_Traffic
        Anomalous_Traffic
            Abnormal_Traffic
            VoIP_Attack_Traffic
                Abuse_of_Service_Attack
                Disruption_of_Service_Attack
                Interception_and_Modification_of_
                Man-In-the-Middle_attack
                SIP_Invite_Flooding_Attack
                SIP_Malformed_Message
                Social_Attack
                Spam_Over_Internet_Telephony
        Normal_Traffic
```

Query (class expression): Interception_and_Modification_of_Service_Attack

Query results – Sub classes (1): Man-In-the-Middle_attack

Figure 4.4: Query 1:Find an attack that belong to the group of Interception and Modification attack

and correctness) of rule database before system implementation is very important. It avoids the risk of deploying ineffective intrusion detection systems consist of models with inconsistent assumptions. Fig. 4.6 shows the model of anomaly detection system before reasoning. It is noticed that the attacks are not organized according to their groups. While, Fig. 4.7 shows the same model after consistency and satisfiability checking after reasoning with semantic reasoner Pallet [138]. It shows that the model is consistence and the concepts (e.g., attacks) are organized according to their groups.



```
IsIdentifiedBy only Disruption_of_Service_Attack
```

Is_Message_Format_Correct
Number_of_Invite_Message

Figure 4.5: Query 2: Find Two possible features for detecting INVITE Flooding Denial of Service attacks

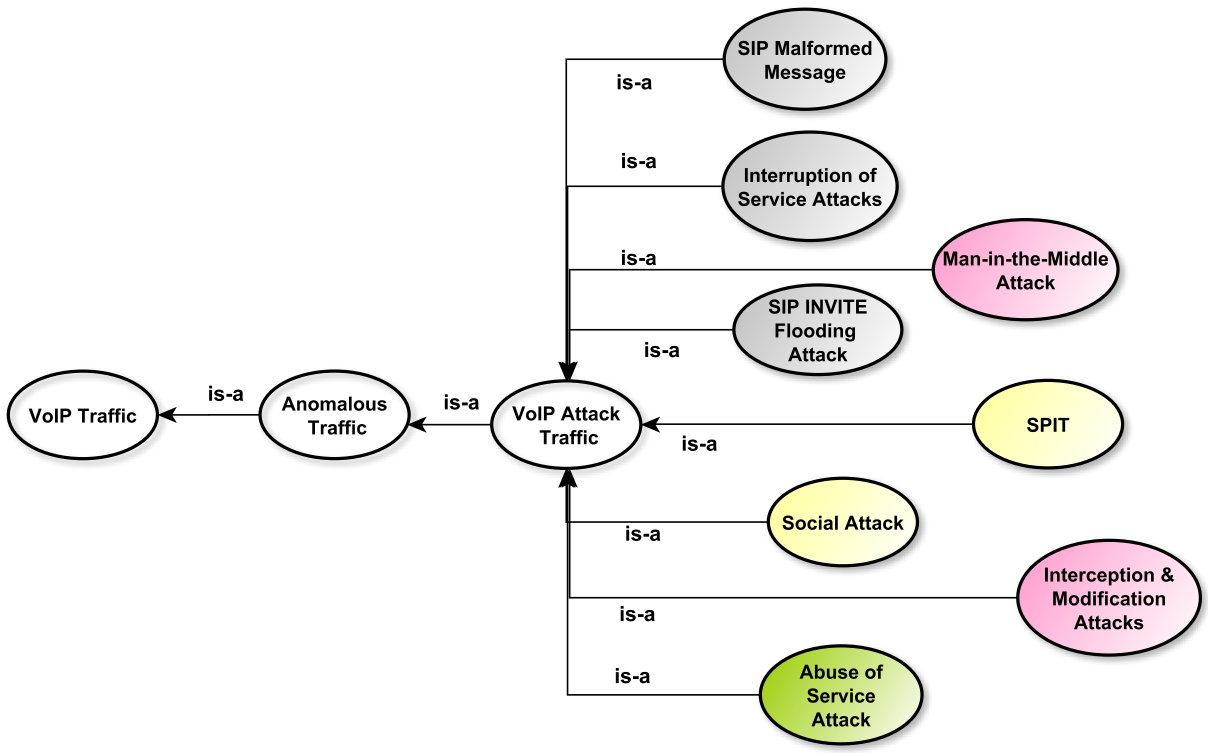


Figure 4.6: VoIP-ONTO : Taxonomy before reasoning

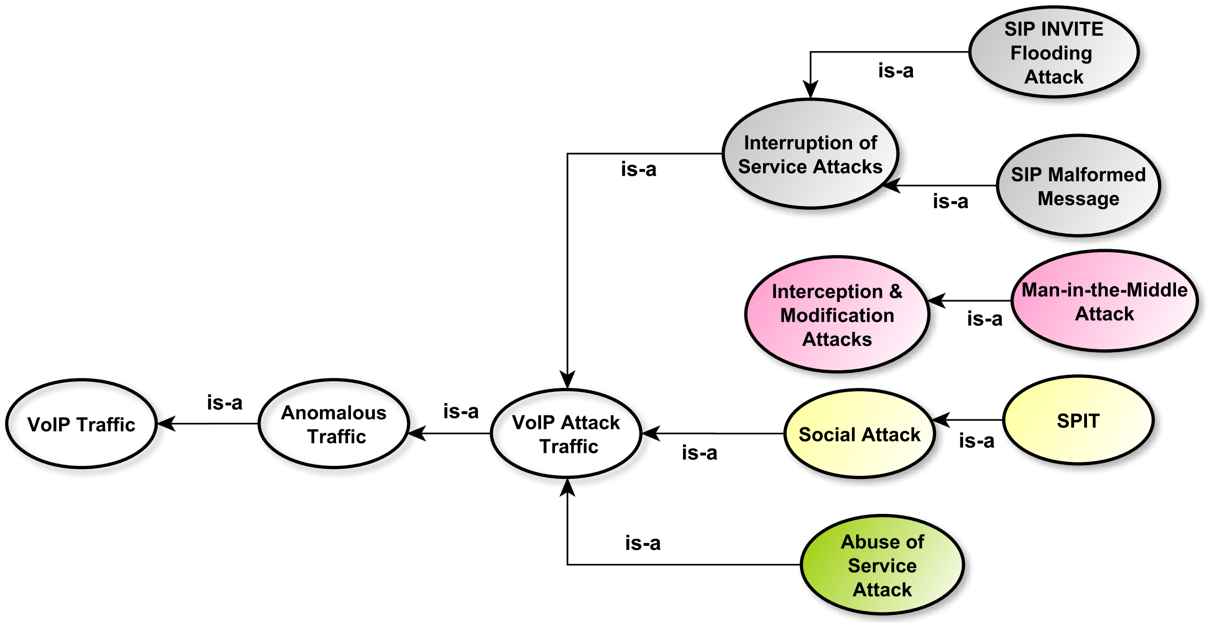


Figure 4.7: VoIP-ONTO : Taxonomy after reasoning

Chapter 5

Traffic Generation, Collection and Anonymization

In recent years researchers have designed and implemented various security testing tools, classifiers and control systems for the analysis and protection of SIP based services. To perform the analysis of SIP services and to test the efficiency of these classifiers and control systems, a large number of SIP trace is required. Unfortunately, to obtain real-world SIP traces of attack incidents or even normal traffic is difficult as VoIP providers are not willing to distribute their data due to user privacy agreements. No VoIP data corpus is publicly available such as repositories[1] that maintain data sets as a service to the machine learning and data mining community. Lack of a publicly available dataset containing VoIP traces also hinders the possibility of comparing the results of different security control systems.

In this work, the analysis of the SIP based VoIP service and testing of our filtering system are performed by collected SIP traces from our institution. The system is up and running since May 2011. Since then,

[1]UC Irvine Machine Learning Repository, http://archive.ics.uci.edu/ml/,
Datasets for Data Mining,
available at: http://www.inf.ed.ac.uk/teaching/courses/dme/html/datasets0405.html,
DARPA Intrusion Detection Data Sets,
http://www.ll.mit.edu/mission/communications/cyber/CSTcorpora/ideval/data/

more than 242 million SIP messages have been analyzed.

5.1 Traffic Collection & Anonymization

To collect real SIP traffic, we have established an agreement with our institution[2] that allows the collection of SIP traces by mirroring the port in front of the SIP Proxy server. The mirror port monitors and captures all the SIP messages (associated with the incoming and outgoing calls) that pass the SIP servers. Before storing the captured SIP traces into the local storage, SIP messages are anonymized to hide all the sensitive information available in a SIP message in order to mitigate privacy and security concerns and to comply with the legal requirements [85].

The process of capturing and anonymization of SIP traffic is performed in pipeline fashion. The monitoring process collects SIP messages in the server and stores in a temporary buffer in the same machine where monitoring process is running. For the anonymization purpose, a script, containing codes for anonymization, is used. It is activated by a time driven trigger. The anonymization script, thus activated automatically after a regular interval (here, 24 hours), anonymized the file (stored in the temporary buffer) that contains the captured SIP packets for the last 24 hours. The file containing anonymized SIP messages is then transferred to our local storage for analysis and the temporary buffer is cleared to make room for new captured packets.

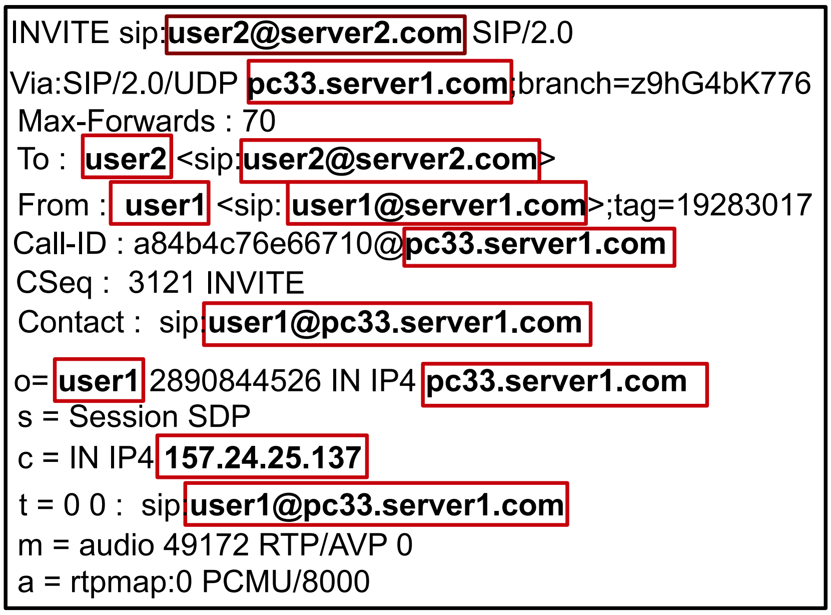
5.1.1 Sensitive Information in a SIP Message

A SIP message is either a request from a client to a server or a response from a server to a client. The "request line" specifies the type of request being issued, while the "response line" indicates the success or failure of a request.

[2]SISTI office (Sistemi Informativi, Servizi e Tecnologie Informatiche), University of Trento

Both the request and the response contain a "start-line" (request/response line) followed by one or more headers and a message body. Example of a SIP INVITE request message is shown in Fig. 1.2.

Sensitive information in a SIP Message refers to any kind of knowledge about SIP user agents and servers in a SIP message that prevents the distribution of that message due to privacy and security issue. Fig. 5.1 highlights the fields containing sensitive information in a SIP INVITE message. Fields containing information about the service provider, ser-



```
INVITE sip:user2@server2.com SIP/2.0
Via:SIP/2.0/UDP pc33.server1.com;branch=z9hG4bK776
 Max-Forwards : 70
 To : user2 <sip:user2@server2.com>
 From : user1 <sip: user1@server1.com>;tag=19283017
Call-ID : a84b4c76e66710@pc33.server1.com
CSeq : 3121 INVITE
 Contact : sip:user1@pc33.server1.com

o= user1 2890844526 IN IP4 pc33.server1.com
 s = Session SDP
 c = IN IP4 157.24.25.137
 t = 0 0 : sip:user1@pc33.server1.com
 m = audio 49172 RTP/AVP 0
 a = rtpmap:0 PCMU/8000
```

Figure 5.1: Sensivite Information in a SIP INVITE Request Message.

vice route, server, "caller" or "callee" (such as, SIP Uris, IP address) in a SIP message are described below together with `example of usage`.

- **Request line**–The first line of a SIP request message containing information about request method and protocol version, and, address of the "callee".

  `INVITE sip:user2@server2.com SIP/2.0`

- **Contact**–Contain a display name, a URI with URI parameters, and

header parameters.
`Contact:< sip:alice@server3.com>`

- **From**–Indicates the initiator of the request.
  `From:Caller <sip:caller@server1.com>`

- **To**–Specifies the logical recipient of the request.
  `To:receiver <sip:user2@server2.edu>`

- **Via**–Indicates the path taken by the request and also should be followed in routing responses.
  `Via:SIP/2.0/UDP server1.com:5060;branch=z9hG4bK87asdks7`

- **Authorization**–Contains authentication credentials of a user agent.
  `Authorization:Digest username=‘‘user1’’,realm=‘‘server1.com’’`

- **Call-Info**–Provides additional information about the caller or callee.
  `Call-Info:<http://wwww.example.com/user1/photo>`

- **In-Reply-To**–Enumerates the Call-IDs that a call references or returns.
  `In-Reply-To:70710@server1.com`

- **Organization**–Conveys the name of the organization to which the SIP element belongs. `Organization:organization_name`

- **Proxy-Authenticate**–Contains an authentication challenge.
  `Proxy-Authenticate:Digest realm=‘server1.com’’,`
  `domain=‘sip:ss1.carrier.com’’,algorithm=MD5`

- **Proxy-Authorization**–Contains client’s identity (to proxy) that requires authentication.
  `Proxy-Authorization:Digest username=‘‘user1’’,`
  `realm=‘‘server1.com’’`

- **Reply-To**–Contains a logical return URI that may be different from the From header field.
  `Reply-To:Bob<sip:user3@server1.com>`

- **Route**–Contains list of proxies used to force routing for a request.
  `Route:<sip:server1.com; lr>`

5.1.2 Anonymization of SIP messages

A script performs the anonymization of SIP messages. This script takes individual SIP message as an input and anonymizes all the sensitive information available. Particularly, it hides all the *SIP URIs* and *IP addresses* so that identity of the "caller" and "callee" is not disclosed.

The anonymization is performed using a cryptographic hash function which is one of the most popular methods of anonymizing data. We have

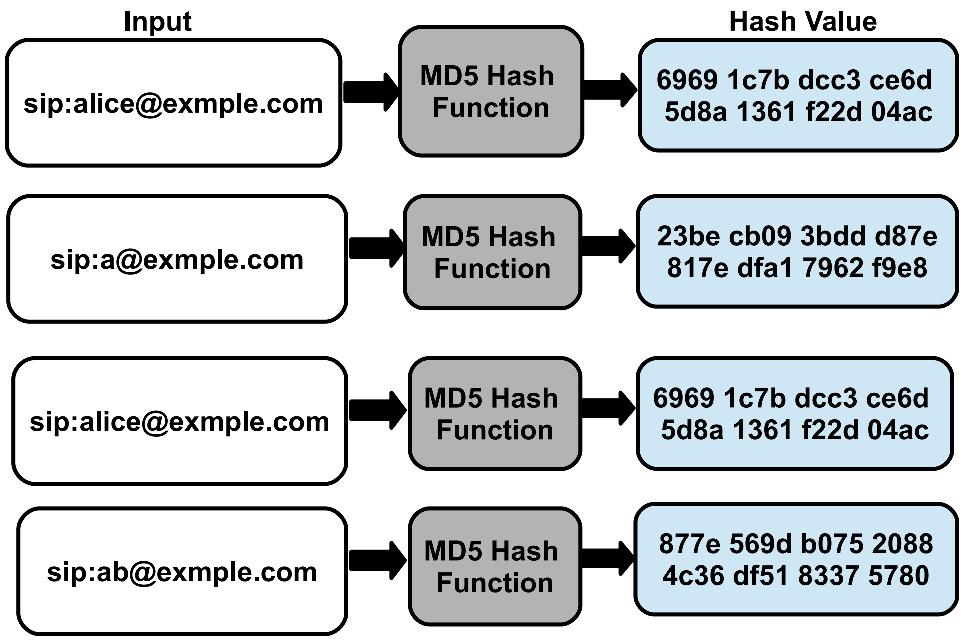

Input
Hash Value
sip:alice@exmple.com
MD5 Hash Function
6969 1c7b dcc3 ce6d 5d8a 1361 f22d 04ac
sip:a@exmple.com
MD5 Hash Function
23be cb09 3bdd d87e 817e dfa1 7962 f9e8
sip:alice@exmple.com
MD5 Hash Function
6969 1c7b dcc3 ce6d 5d8a 1361 f22d 04ac
sip:ab@exmple.com
MD5 Hash Function
877e 569d b075 2088 4c36 df51 8337 5780

Figure 5.2: Example of a data anonymization using hash table.

used the MD5 message-digest algorithm [153] which has been utilized in a wide variety of cryptographic applications, and also for verifying data integrity. MD5 hash function produces a 128-bit (16-byte) hash value, typically expressed in the text format as a 32 digit hexadecimal number.

In the anonymization process, each occurrence of a SIP URI and a IP address is hashed. An example of the hashing process of SIP URIs is shown in Fig. 5.2.

5.2 "SIP-Msg-Gen" - Synthetic SIP Messages Generator

As could be expected on a closed network such as ours, there were no "bad" messages that were maliciously sent with the intention of creating disruptions of service or of harming the network[3]. To counteract the lack of "bad" traces in our collected sample, we focus on an alternative approach which is to synthetically generate "bad" traces and randomly inject them into the stream of real "good" VoIP traces.

To mitigate the lack of *syntax* and *semantic* errors in our collected sample, we developed "SIP-Msg-Gen"[4], a synthetic SIP message generator, capable of generating both "good" and "bad" SIP messages. "SIP-Msg-Gen" is available under GPL license terms. It is a stateful and context aware simulator consists of a *syntax fuzzer* sepcifying the syntax to generate a well-formed SIP message and rules to manipulate it, and a set of *state transaction rules* to model the transitions of messages in the system. State transition rules are defined based on the test scenarios, where a scenario represents a high level goal. Test scenarios are: (i) a basic call flow between two user agents, (ii) a basic call flow between two user agents with

[3]This condition is however about to change as our institution is going to be directly connected with public VoIP networks. This work is an attempt to prepare ourselves for protecting the network from malicious users in such public environment.

[4]"SIP-Msg-Gen", available at: `http://disi.unitn.it/~ferdous/SIP-Msg-Gen.html`

fuzzed SIP messages, (iii) a SIP user agent is under INVITE flooding attack, and (iv) a SIP user agent is under REGISTER flooding attack. The generation of SIP messages begins with the automatic generation of N SIP URIs. Only identifiers are generated, not instances of a user entity, so that configuration of users characteristics is not required.

5.2.1 Syntax Fuzzer

SIP is a text-based protocol, the information exchanged within the protocol is encoded as strings of characters. Syntax of the SIP messages are defined by a context-free grammar specified in Augmented Backus-Naur Form (ABNF) [35], a metalanguage based on Backus-Naur Form (BNF). Context free grammars such as ABNF are effective in generating test data [120]. Our simulator is able to flexibly generate thousands of well-formed SIP messages following the ABNF syntax of SIP messages defined in the IETF standard.

Fuzzing of SIP Messages

A fuzzed SIP message can be either a valid "torture" message to stress the SIP server or an invalid message that does not comply with the syntax or the semantic of SIP protocol specification. In the context of the SIP protocol, there are endless ways to fuzz a SIP message. This can be done either by crafting an well-formed SIP message or generating a *bad* SIP messages from the scratch by defining a comprehensive grammar for fuzzed SIP messages. We follow the first option where the idea is to tamper a well-formed SIP message by injecting invalid, unexpected, or random data. Generation of a SIP message with *syntax* and *semantic* error is inspired by "SIP torture test" as defined in RFC 4475 [177] and PROTOS test suite [201]. The "malicious" messages are generated following the attack

Table 5.1: Types of fuzzed SIP messages

Malformed Messages	
Error in Request line	
Syntax error in one/multiple mandatory header field	
Syntax error in one/multiple optional header field	
Syntax Error in body of a message	
Crooked Messages	
Missing mandatory fields	
Duplicate entry for unique fields	
Unknown/Invalid Protocol version	
Presence of garbage string/invalid character in message	
Hierarchical disorder of message structure	
Unknown scheme for header fields and message body	
Unknown Method Name and Response status	
Tortured Messages	
Overlarge Message Body	
Extra white spaces between colons, semicolons, header field values	
Use of extra colons, semicolons in header fields	
Use of Escaped characters in messages	
Presence of empty optional header fields and parameters	
Line folding all over the message	

scenarios described in [101, 131, 184]. Table 5.1 shows the scenarios that the simulator considers to fuzz a SIP messages.

5.2.2 Test Case:Generation of a basic call

This test case follows a very basic SIP call flow between two user agents Fig. 5.3 shows the state transition diagram followed by "SIP-Msg-Gen" to generate SIP messages involved in a call flow (illustrated in subsection 1.2.5 in Fig. 1.3) between two user agents. The model starts at the *INIT* state, when a user agent A (caller) sends an INVITE message indicationg that A wish to communicate/talk with othe user agent B (callee). Here, A and B are randomly selected two SIP URIs from the pre-defined user of SIP URIs by the simulator. At this stage, "SIP-Msg-Gen" generates a

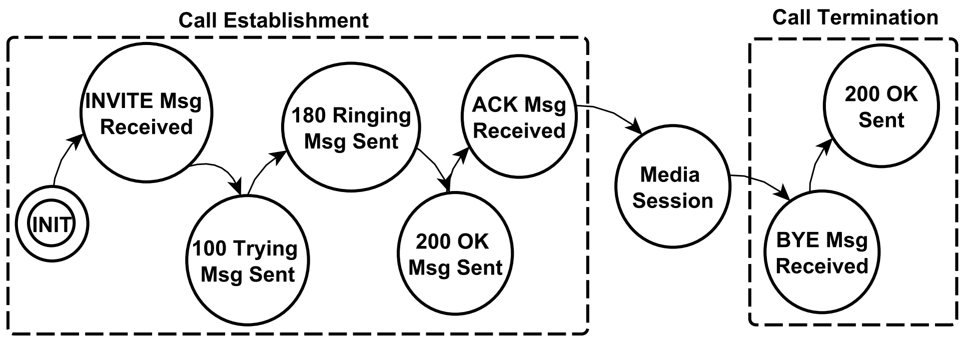


Figure 5.3: State transition diagram of a basic call flow between two SIP user agents.

well-formed SIP INVITE message and transits to the next state *INVITE Msg Received*. Upon receiving the INVITE message the next state is a response message (TRYING 100) sent by the proxy server to UAC A to acknowledge that the INVITE is being processed. The UAC B sends a response (Ringing 180) when his telephone begins to ring. The state of the call establishment is performed by UAC B by responding (OK 200 message) to the invitation and a connection is established. Then media streams are exchanged directly between UAC A (caller) and UAC B (callee). The session is ended by an BYE request message to the server from any UAC (A or B) and a repliy with an OK 200 message from other UAC.

5.2.3 Test Case:Presence of fuzzed SIP message in a call flow

Fig. 5.4 shows the state transition diagram followed by "SIP-Msg-Gen" to generate a basic call flow between two user agents where the SIP messages involved in the dialog can be fuzzed. At the *INIT* state, the predicate checks the value of the boolean variable *Fuzz Flag*. *Fuzz Flag* assigned as *true* refers to a transition to state *Fuzzed INVITE Msg* where a fuzzy SIP INVITE message is generated by the simulator. On the other hand, *Fuzz Flag* assigned as *false* indicates the transition to state *INVITE Msg* where an well-formed SIP INVITE message is generated. This above process is

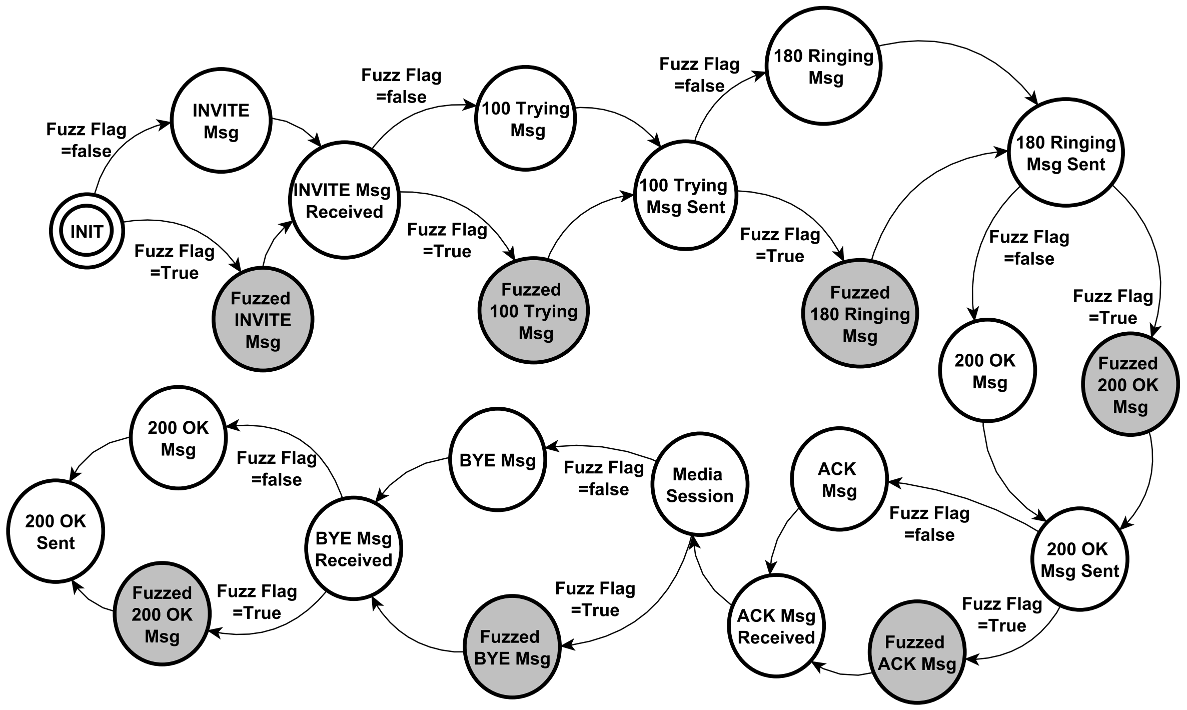


Figure 5.4: State transition diagram of a basic call between two SIP user agents using fuzzed SIP messages.

followed through all the states involved in call establishment and termination. The value (e.g., true and false) of the *Fuzz Flag* is selected randomly in every state.

5.2.4 Test Case:SIP INVITE flooding attack scenario

Fig. 5.5 shows the state transition diagram for generating the INVITE request flooding attacks. This model is inspired by the state transition model defined in [167]. The model starts at the *INIT* state with the generation of a SIP message. At this state, the predicate checks the message type of the received message. On receiving the first INVITE request, the state machine makes a transition from the *INIT* state to the intermediate state *INVITE Msg Received*. During this transition, a counter *packet_counter* is started

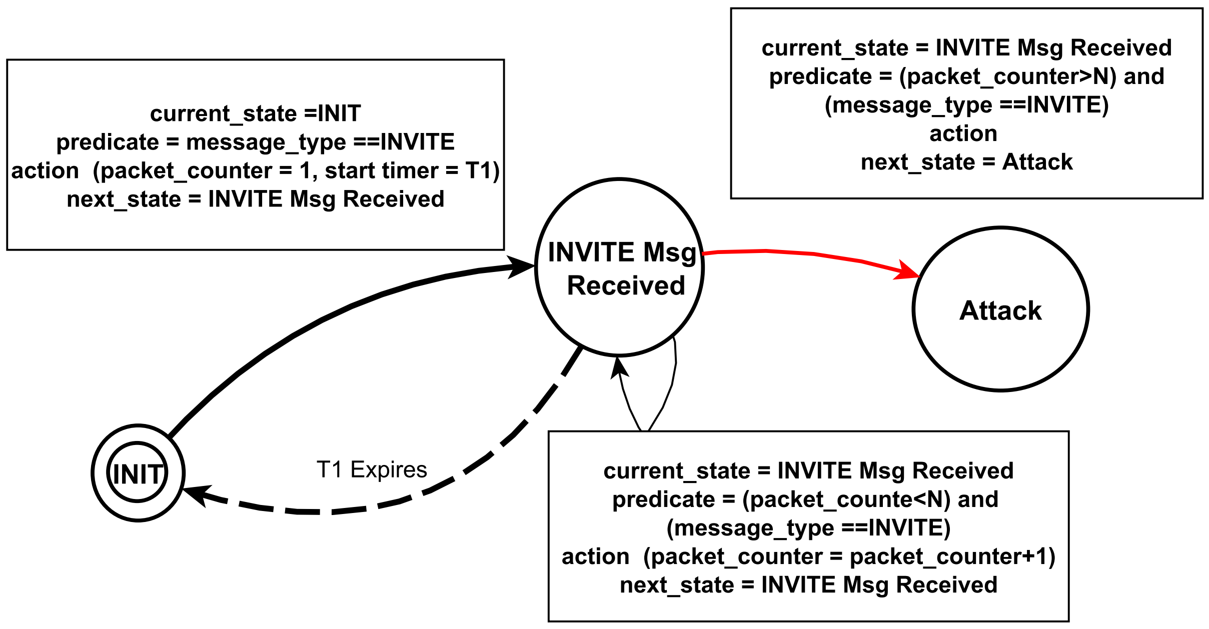

current_state =INIT
predicate = message_type ==INVITE
action (packet_counter = 1, start timer = T1)
next_state = INVITE Msg Received
current_state = INVITE Msg Received
predicate = (packet_counter>N) and
(message_type ==INVITE)
action
next_state = Attack
INVITE Msg Received
Attack
INIT
T1 Expires
current_state = INVITE Msg Received
predicate = (packet_counte<N) and
(message_type ==INVITE)
action (packet_counter = packet_counter+1)
next_state = INVITE Msg Received

Figure 5.5: State transition diagram of SIP INVITE flooding attack.

to count the received INVITE messages for the same destination within a certain amount of time (T_1). Timer T_1 sets the time window, under which N received INVITE requests are considered as normal. To immitate the flooding attack scenario "SIP-Msg-Gen" generates (N*2) INVITE request during time window T_1.

5.3 VoIPTG - VoIP Traffic Generator

In this thesis, we also concentrate on detecting the *malicious* intruders by training a classifier with the knowledge about the social behavioral pattern of *legitimate* VoIP users. For the purpose of evaluating the performance of the classifier, a huge amount of high level VoIP traffic (e.g., call detail records) containing *malicious* content is required. Our collected SIP traces do not contain any information about *malicious* users due to the lack of "bad" messages that were maliciously sent with the intention of harming the network. To counteract this limitation, we have developed “VoIPTG” (VoIP Traffic Generator)[5], a flexible and generic traffic simulator, which is capable to generate traffic following various possible models of SIP based VoIP system with special attention to characterize the more sophisticated user behavior in the system. “VoIPTG” is available under GPL license terms. One added bonus of the current version of “VoIPTG” is that it is capable to generate a combination of “good” and “bad” traffic instead of only “good” or only “bad”.

5.3.1 State-of-the-art of Synthetic traffic Generation

Designing models for generating synthetic traffic of SIP based VoIP system is not trivial as SIP is one of the most complex and hard to debug IETF standard ever. Its extensibility, the fact that it must maintain an often complex status of the session, the structure of proxy agents and servers that define the global organization of services like telephony, conferencing, and so forth, indicates that characteristic of traffic in SIP based VoIP services is different from traditional fixed telephony networks. Difference in characteristics also means different parameters to characterize the traffic, thus, traditional traffic models such as the Erlang, Engset and Poisson model are not sufficient to model SIP based VoIP traffic [30, 36, 37, 77]. This introduces the necessity for a new consideration on modeling the traffic of the SIP based VoIP services.

Though many projects are focusing on developing tools and solutions for generating synthetic telephony traffic, there is very little available in terms of simulator for generating traces of SIP based VoIP system. Moreover, lack of flexibility in deployment and sufficient guidance to use limit the scope of this simulators.

[5]Source code and user manual of “VoIPTG” is available at - `http://disi.unitn.it/~ferdous/VoIPTG.html`

A list of VoIP security testing tools and attack simulators is found in [206]. Authors of [48] mention a list of attack simulators to test the security threats of VoIP applications.

“SIPp” [172] is a free open source traffic generator for the SIP protocol. It includes a few basic user agent scenarios (UAC and UAS) to establish and release multiple calls with the INVITE and BYE methods. It generates traffic by repeating call flow scenarios described in custom XML. Though it allows the definition of call flows from very simple to complex one using XML, defining a large number of call flows to generate a huge amount of SIP traffic is tedious and time consuming.

It is noticed that, the tools mentioned above follow simple techniques to generate random SIP messages/calls instead of considering the complex traffic models, hense, they are mainly used for testing the robustness of SIP implementation. Moreover, these available tools do not identify all the key parameters required for characterizing the SIP based VoIP traffic and fail to capture the complex statistical properties of the VoIP traces. More explicitly, these implementations ignore the complex user behavior in the SIP based services which is very significant due to the unpredictable behavior of users in the network.

Authors of [93] propose a conversational model based VoIP traffic generation algorithm, which simulates the behavior of two users in a VoIP session. This paper considers the “on-off” pattern of VoIP traffic [49, 79] where voice traffic can be characterized by a succession of active periods (talkspurt or “on”) followed by inactive period (silence or “off”). According to this behavior of speakers, this paper proposes a conversational model to simulate two speakers conversational state change and interactive process.

A framework to annotate and customize VoIP traffic considering the different nature of traffic and user behavior in SIP based VoIP application is proposed in [193]. Normal traffic is generated by following a traffic model where the call arrival pattern follows a poisson distribution and the call holding time follows an exponential distribution. The simulator provides the opportunity to define a social model for the set of users. Here, the social model of users follows a bimodal distribution where represents the probability that a user calls another socially connected user is P_{social}, while $(1 - P_{social})$ is the probability that a user calls another socially not/less connected user. The simulator generates attack traffic using currently available VoIP assessment tools such as “inviteflood”, Sipscan and “Spitter”. Though [193] and [93] focus on modeling the user behavior in synthetic traffic generation, additional architectural decisions is required for the stochastic simulator to generate more realistic and interactive synthetic traffic. This is due to the fact that the real life scenario may be too complex to model with simple statistical distributions. Explicitely, the call duration and volume per user distribution of SIP based VoIP traffic follow very complicated models. [30] shows that VoIP CHT (Call Holding Time) distribution is not exponential as traditional telephony and it can be accurately approximated by a mix of two log-normal distributions. Again [37] shows that call holding times follow heavy-tailed distribution rather than exponential distribution and it proposes the generalized Pareto distribution for modeling the call holding times. [181] shows that the call distribution over the users in SIP based VoIP has a power-law property which can be characterized with Zipf Distribution. This introduces the necessity of implementing a stochastic generator that is capable to model the realistic behavior profiles for users and attackers, and, provide flexibility and efficiency in generation of large amount of synthetic traffic of SIP based VoIP system. Considering this situation, we concentrate on developing a synthetic generator for generating traces of SIP based VoIP services.

5.3.2 Traffic Models of VoIPTG

As mentioned before, analysis on VoIP traces shows the difference in the characteristics of VoIP traces from the traditional POTS and ISDN. With the rapid change of telephony technology from traditional PSTN to SIP based VoIP architecture, it is expected that the ongoing transition toward IP telephony will also change the characteristics to define the traffic [80]. Characterization of SIP based VoIP traffic with one specific model is difficult as the traces can be very different depending on the network structure, vendors and users.

Traditional telephony traffic monitoring was mainly focused on *interarrival times* and *call holding times* (the time from the callee answer to the end of the circuit tear-down) in order to properly dimension the network and estimate revenues. In modern IP-telephony, network dimensioning is less of a concern: capacity in IP-based systems is cheap (at least for voice-related traffic volumes), and SIP-based signaling can support millions of users on off-the-shelf hardware. All the three points above hints the same direction for metrics of interest for the new telephony: a) call durations; b) ringing times; c) volume of generated traffic per address/user over a certain time-span; and d) multiple active calls per address/user. Many additional metrics can be found and may be of interest, but the four above are fundamental, thus before considering any other it is worth exploring the importance and characteristics of these four. Metrics a) and b) can be associated to calls without a per-user architecture. Metrics c) and d) are instead strictly correlated to the users' behavior. Table 5.2 presents the list of possible distributions/patterns for the four above parameters that current implementation of "VoIPTG" is able to follow in order to generate SIP traffic.

Parameter	Implemented Distribution
Call Duration	Exponential Distribution, Log Normal Distribution
Interarrival Time	Exponential Distribution
Volume per Caller	Call generation with Delta rate, Uniform(Max-Min), Zipf Distribution
Volume per Receiver	Call receive with Delta rate

Table 5.2: [
Distributions for fundamental parameters for generating synthetic VoIP Traffic]List of implemented distributions for fundamental parameters for generating synthetic VoIP Traffic.

5.3.3 VoIPTG - Logical Architecture

With the suitable choice of models of network parameters (listed in Table 5.2), "VoIPTG" is able to generate trace that immitates the comple real world SIP based VoIP traffic. Fig. 5.6 reports the logical flow of call generation in "VoIPTG".

The generation of a trace begins with the automatic generation of N_u users identifiers. Only identifiers are generated, not instances of a user entity, so that configuration of users characteristics is not required. The generation of user domain is important in this context as "VoIPTG" pays special attention on characterizing the sophisticated behavior of users in a network. VoIPTG maintains two user sets to indicate the *caller* set who generates the call and the *receiver* set who receives the generated call. These two user sets are not disjoint and the overlapping ratio between these two user sets is configurable. This concept can be extended to the concept of *internal* and *external* user of a VoIP service provider.

Once the users' set is created, call generation follows a streamlined procedure: a new call is characterized by its (global) interarrival time from the preceding one, its call duration, and finally it is assigned to one of the users' identifiers available as caller and one as callee. Shaded block in Fig. 5.6 are the stochastic characterization of calls and they are inde-

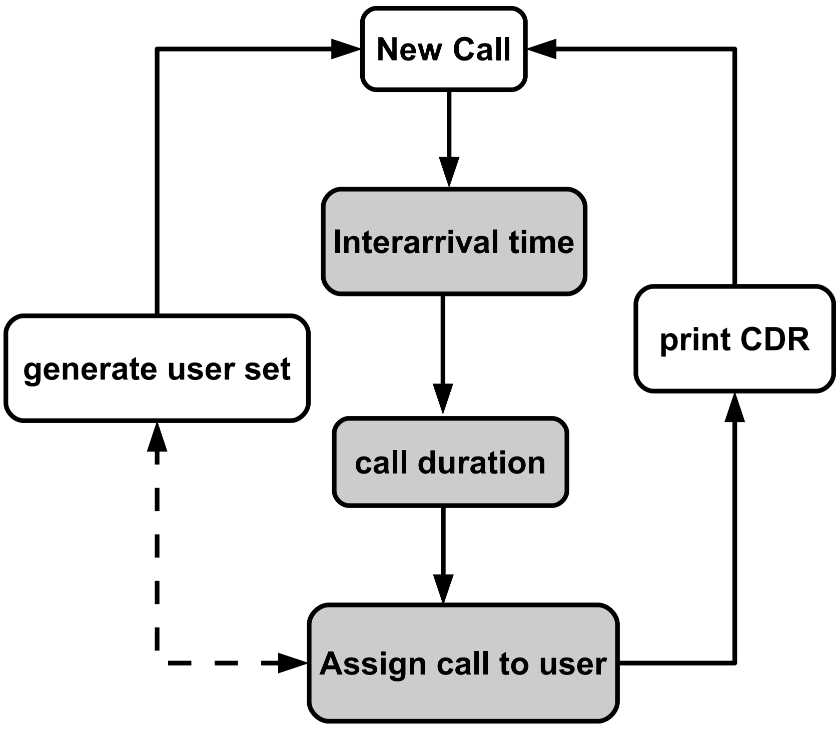


Figure 5.6: Work flow of Synthetic Traffic Generator(VoIPTG).

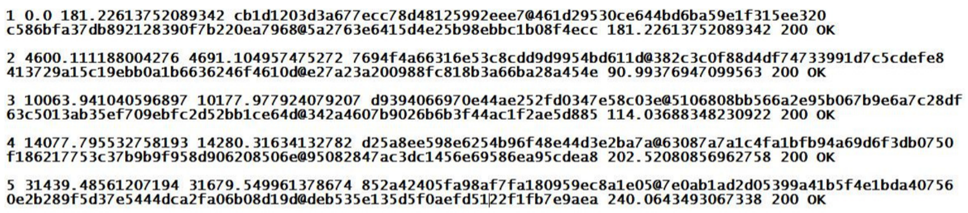
pendent one another and independent from the users identifiers, so that "VoIPTG" configuration is simple and call generation is fast and efficient. Any stochastic block can take different distributions as listed in Table 5.2. The dashed connection between the block generating the user set and the block assigning calls to the users indicates only sharing of the database and not a logical flow in the generation process. "VoIPTG" is written in Java and generates in the order of 10^6 calls per second on standard, off-the-shelf hardware.

The input of "VoIPTG" is the configuration file contains the input parameters of the traffic generation and the output of the simulator is a list of calls refers to communication between two SIP users for the purposes of a multimedia conversation. The generated calls are represented in the form of standard Call Details Record (CDR) which can be read by any call

Call Info	**Details**
Call ID	Sequence number of the generated call(long number)
Start Time	Start time of the call (second)
End Time	End time of the call (second)
Call Duration	Duration of the call (second)
Caller	User from user domain (sip address)
Receiver	User from user domain (sip address)
Response	"OK"
Response Code	200 (Code 200 indicates the established call)

Table 5.3: Format of a synthetic call detail record by “VoIPTG”

monitoring system where each call is identified with a Call ID. Each CDR is also associated with detail information such as call duration, starting time and ending time of the call, caller and receiver. The output format of VoIPTG is found in Table 5.3 and Fig. 5.7 shows an example of output file containing synthetic call records.



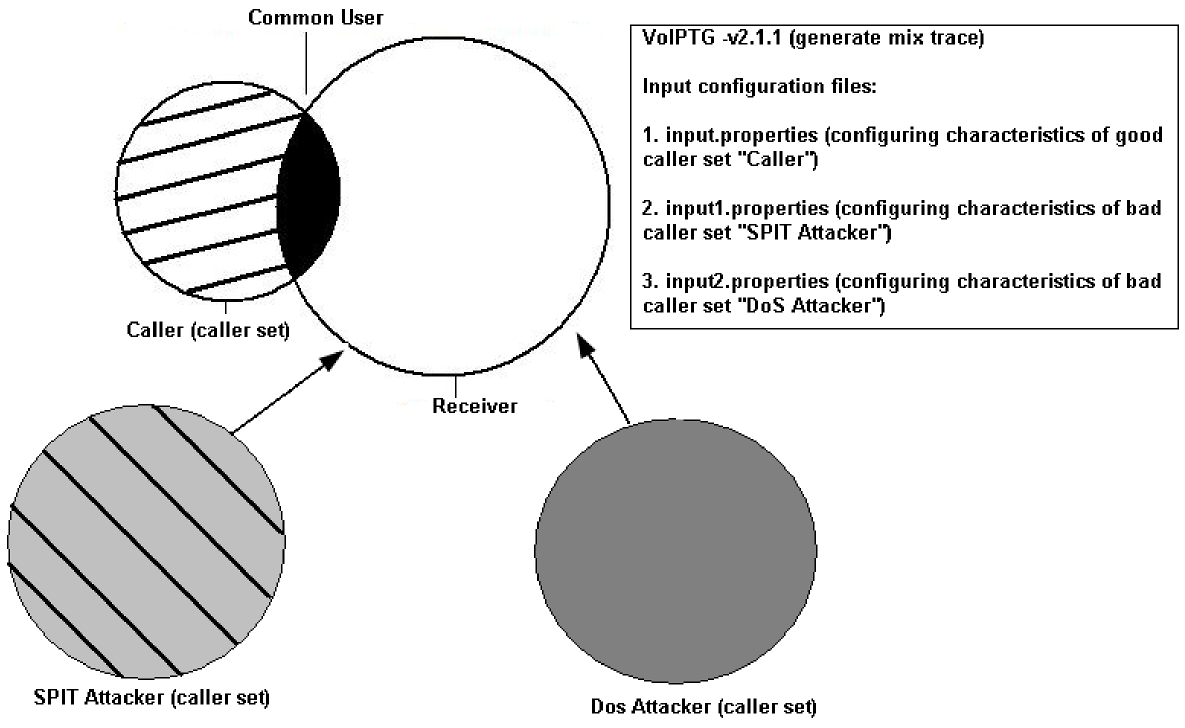
```
1 0.0 181.22613752089342 cb1d1203d3a677ecc78d48125992eee7@461d29530ce644bd6ba59e1f315ee320
c586bfa37db892128390f7b220ea7968@5a2763e6415d4e25b98ebbc1b08f4ecc 181.22613752089342 200 OK

2 4600.111188004276 4691.104957475272 7694f4a66316e53c8cdd9d9954bd611d@382c3c0f88d4df74733991d7c5cdefe8
413729a15c19ebb0a1b6636246f4610d@e27a23a200988fc818b3a66ba28a454e 90.99376947099563 200 OK

3 10063.941040596897 10177.977924079207 d9394066970e44ae252fd0347e58c03e@5106808bb566a2e95b067b9e6a7c28df
63c5013ab35ef709ebfc2d52bb1ce64d@342a4607b9026b6b3f44ac1f2ae5d885 114.03688348230922 200 OK

4 14077.795532758193 14280.31634132782 d25a8ee598e6254b96f48e44d3e2ba7a@63087a7a1c4fa1bfb94a69d6f3db0750
f186217753c37b9b9f958d906208506e@95082847ac3dc1456e69586ea95cdea8 202.52080856962758 200 OK

5 31439.48561207194 31679.549961378674 852a42405fa98af7fa180959ec8a1e05@7e0ab1ad2d05399a41b5f4e1bda40756
0e2b289f5d37e5444dca2fa06b08d19d@deb535e135d5f0aefd5122f1fb7e9aea 240.0643493067338 200 OK
```

Figure 5.7: An example of output Call Detail Records of “VoIPTG”.

5.3.4 Example - Synthetic Trace Generation with “VoIPTG”

The current version of “VoIPTG” provides the flexibility of defining multiple caller sets with different call generation property. This feature can be used to replicate the real world telephony scenario where in the stream of “good” trace there is the existence of “bad” trace. The “good” traces are generated by *legitimate* users such as , home users, commercial users, etc. while “bad” traces are the attacks from *malicious* users.

This sub-section shows an example of generation of mixed (a combination of "good" and "bad") synthetic traces with "VoIPTG". In this scenario, we consider the user domain of a telecom provider. Its *legitimate* user domain can be divided into two overlapping sets defined as *Caller* and *Receiver*. Again, there may be some attackers outside the network of the telecom provider, who generate spam calls (SPIT) or flooded the server with calls to disrupt the service. These callers are gathered into two sets defined as *SPIT Attacker* and *DoS Attacker*. Call generated by the set of *Caller*, *SPIT Attacker* and *DoS Attacker* are directed to the users of the set of *Receiver*. Among these, calls generated from users from *Caller* set are defined as good calls. This above described user domain is presented in Fig. 5.8.



VoIPTG -v2.1.1 (generate mix trace)

Input configuration files:

1. input.properties (configuring characteristics of good caller set "Caller")

2. input1.properties (configuring characteristics of bad caller set "SPIT Attacker")

3. input2.properties (configuring characteristics of bad caller set "DoS Attacker")

Figure 5.8: Example of User domain for synthetic traffic generation.

Fig. 5.8 shows that to model the above scenario, three configuration files (named *input.properties*, *input1.properties* and *input2.properties* are prepared considering the call generation patterns of three caller sets (e.g., *Caller*, *SPIT Attacker* and *DoS Attacker*).

The generation of traffic starts by creating the three caller sets and the universal receiver set. “VoIPTG” instantiates new traffic generation phases according to the configuration files containing call generation property for each caller set. Finally the generated traces are combined together in a file and thus we get the merged file which is a combination of “good” and “bad” traffic.

Chapter 6

Conformance and Security Analysis

This chapter reports the conformance and security analysis of SIP messages. Part of this work is published in [56] and [57]. In this work, SIP messages are analyzed to be classified as "good" or "bad" depending on whether their structure and content are deemed acceptable or not. The set of "bad" messages contains those whose *syntax* is incorrect and those whose *semantics* can cause problems. In addition, there are messages that per-se are *syntactically* and *semantically* correct but that collectively can pose a threat to the system. Correspondingly, in Section 3.2, we label messages in the three sets as *malformed*, *crooked* and *malicious* messages.

6.1 Filtering Methodology

Because of the different structure, contents and timing of the SIP "bad" messages, their classification is best carried out by specialized detectors. Briefly, in this work the message analysis is carried out by a multistage classifier: the first part consists of a pretty straightforward lexical analyzer that checks if the message obeyed the grammar of the SIP protocol stated in [156]. As a by-product of the lexical analysis, a sequence of features are extracted from the stream of messages. These features are analyzed for the second stage in which a Support Vector Machine (SVM) [192] after a

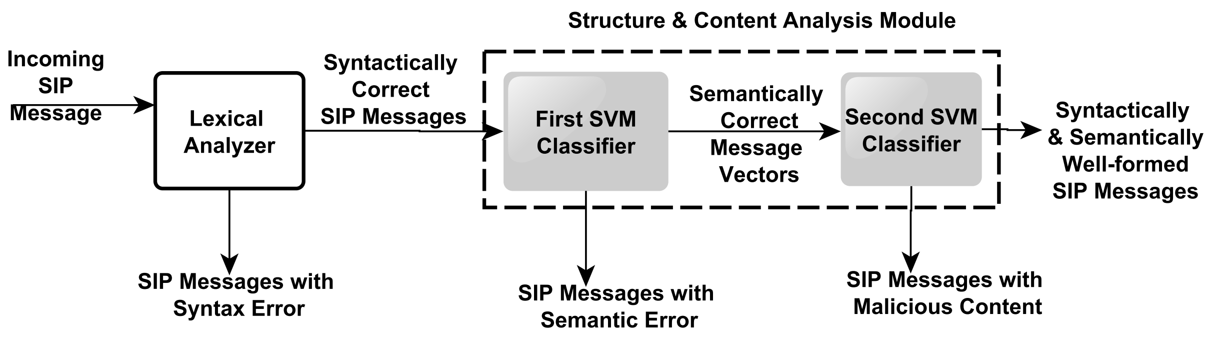

Structure & Content Analysis Module
Incoming SIP Message
Lexical Analyzer
Syntactically Correct SIP Messages
First SVM Classifier
Semantically Correct Message Vectors
Second SVM Classifier
Syntactically & Semantically Well-formed SIP Messages
SIP Messages with Syntax Error
SIP Messages with Semantic Error
SIP Messages with Malicious Content

Figure 6.1: Architecture of the SIP message filtering system

suitable training, flags and discards those messages that were classified as *crooked* and *malicious*.

A third phase is added to the multistage classifier to further cull bad messages from those that survived the first two stages. This third stage is also based on an SVM that has been trained to detect *malicious* messages. The overall architecture is shown in Fig. 6.1.

The multistage architecture is maintained here since it allows better separation of the intrinsically different tasks and offers the possibility of pipelined parallelism on separate hardware if message traffic volumes warrant or require classification rates that cannot be achieved by a single processor. Furthermore, multistage SVMs have been considered more effective than single stage SVM in other classification tasks [99, 115].

The first stage follows a deterministic lexical analysis. Given the formal grammar definition of the messages of the SIP protocol (and thus, the syntax of all SIP messages), the implementation of the lexical analyzer can be realized by any of the standard tools (for example, *lex*) that are available under *Unix* to parse a program written in a given programming language. The task here is actually even simpler since, the processing can be stopped immediately upon detection of the first syntax error since there is no need to extract a list of errors present in the whole message. The second and third stage are meant to detect *crooked* and *malicious* messages.

SVM for SIP Message Classification

As mentioned earlier, we have used two separate SVM classifiers to filter *crooked* and *malicious* SIP messages. The syntactically well-formed SIP messages that are successfully parsed by the lexical analyzer are further processed by a specialized kind of tokenization, where the tokens are significant features extracted from the SIP messages; these features are represented as numerical vectors in features space and these vectors are used by the subsequent SVM classifiers. The goal of this feature selection process is to reduce the computational burden and to extract appropriate information as a SIP message contains huge amount of information. These features are selected by expert knowledge about VoIP systems and they take into consideration aspects related to the contents of each message, as well as the distribution in time and kind of a stream of messages.

- **Structure & Content Related**: These features are extracted from individual SIP messages. A list of these features is found in Appendix A. These features characterize the structure and the content of a SIP message (e.g., protocol version, hierarchical structure of a message, frequency of request- line, empty line and mandatory unique header field, value of scalar fields in a message, etc). These features are used to *train* the first SVM classifier for detection of *crooked* messages.

- **Time and Kind Distribution of SIP Message Stream**: While the identification of *syntactically* and *semantically* incorrect SIP messages is possible by controlling individual messages, the detection of *malicious* content is greatly facilitated by examining features that take into account the time and dimension of traces obtained from a SIP message stream.

Features that are extracted from the syntactically well-formed and meaningful SIP messages are listed in Table 6.1. These features are extracted

Table 6.1: List of features characterizing the distribution of SIP message stream

Feature
Percentages of the un-established call at time t.
Percentages of the calls with very low inter-arrival time at time t
Percentages of the calls with high call set up delay at time t
Rejection Ratio - Percentages of the canceled calls at time t
Success Ratio - Percentages of the successful calls at time t
Number of INVITE messages arrived at time t
Number of REGISTER messages arrived at time t
Number of 200 OK response messages arrived at time t
Number of CANCEL messages arrived at time t
Number of BYE messages arrived at time t

from the stream of SIP messages after each δt time interval. Let us define the set of features found in Table 6.1 as X and $X(t) = \{x_1(t), x_2(t), x_3(t), x_4(t), x_5(t)\}$ is the vector of features extracted at time t, while $t = t + \delta t$. Thus, at the beginning while $t = 0$, the set of feature would be $X(0)$. The choice of δt usually depends on the message arrival rate of the SIP-based networks. In this context, we choose $\delta t = 10$ min, while, it can be even sufficiently low (for example: 0.5 sec) for a very large networks.

Two of these features (e.g., percentages of the calls with very low inter-arrival time and expected call that arrived till time t) depends on threshold values (e.g., low threshold of inter-arrival time and number of expected calls at time t). We have defined the suitable values of these thresholds considering the size and message arrival rate of our network.

Training of SVM classifier

We have used the freely available library LibSVM [27] for Support Vector Machines. The SVMs are trained by presenting each SVM with a set of lexically correct SIP messages that have been pre-labeled as -1 (accepted) and $+1$ (rejected).

The training phase determines a configuration for each of the two SVMs that will result in a good classification of subsequent messages contained in a test set. Our experimental results show that SIP message vectors are not linearly separable, so the feature vectors are projected to a higher dimensional space by performing a vector product by a kernel. Care must be exercised in the selection of the specific kernel used and in the choice of some relevant parameters. To this end, experiments are carried out with various kernels and parameters. It is very obvious that the linear kernel do not yield any interesting result and hence the experimentations have focused on polynomial kernels of degree 2, 3, and 4 and on RBF kernels. The latter are reported in the literature as leading to good results and so are deemed preferable despite their higher computational cost. However, in the support documents available with LibSVM it is mentioned that in the case of data with a large number of features (as in our case), sometime a low degree polynomial kernel is preferable. Our experiments confirmed the validity of this suggestion – at least for our specific application.

6.2 Results and Performance

Performance evaluation of any classifier can be established by analyzing the results over a statistically relevant collection of data. In this context, this means that a large number of SIP traces would be needed. We have used both real SIP traces collected from our institution and synthetic traces generated by our developed traffic generator. Details about the traffic collection, anonymization and generation is discussed in Chapter 5.

The first SVM, used for detection of *crooked* messages, is trained with a set of 2000 pre-classified examples (a balanced mix of good and bad messages) where each sample message is transformed into vectors of 26 features. For this classifier, the *test set* contains the unlabeled SIP message

Table 6.2: Description of Real SIP Traces

Description	Number of Msg
Total Message	242,714,093
Message with Syntax Error	2,627
Syntactically well-formed but “bad” message	0
Syntactically well-formed and “good” message	242,711,466

vectors that are *syntactically* well-formed but may contains *semantic* errors. Again, the second SVM classifier, used for detection of *malicious* messages, is trained with a set of 1000 pre-classified examples (a balanced mix of good and bad messages) where each sample message is transformed into vectors of 10 features. For this classifier, the *test set* contains the unlabeled SIP message vectors that are both *syntactically* and *semantically* well-formed, but may contain *malicious* contents.

6.2.1 SIP Dataset

The set of the collected sample from our institution consisting of around two years SIP traces is used here for experimental purpose. Details about the dataset are found in Table 6.2. Out of over 242 millions messages in the data set, only $2,627$ contains syntax errors in optional header fields; all of them are duly rejected by the lexical analyzer. The remaining messages are passed to the first SVM classifier for next level filtering; those messages were all “good” messages and the SVM did not reject any of them.

In the messages collected at our institution, there were no “bad” messages what so ever, which perhaps could be expected. There were neither “crooked” messages, nor “malicious” messages sent with the intention of creating disruptions of service or of harming the network. To counteract the lack of “bad” SIP messages in our collected sample, synthetic “bad” messages generated by “SIP-Msg-Gen” (described in Section 5.2) are injected into the stream of real SIP messages collected at our institution.

Table 6.3: Description of synthetic “malformed”, “crooked” and “tortured” messages in dataset

Synthetic Test Scenarios	**Number of Msg**
Malformed Messages	3,000,000
Syntax Error in “Request/Response Line” of a message	
Syntax Error in header fields of a “Request” message	
Syntax Error in header fields of a “Response” message	
Syntax Error in body of a message	
Null entry for mandatory header fields	
Crooked Messages	5,000,000
Injected SQL code with the aim of gaining unauthorized access	
Multiple “Request/Response Line” in a message	
Unknown/Invalid Protocol version	
Missing mandatory header fields a message	
Duplicate entry for unique header fields	
Presence of garbage string after message body	
Hierarchical disorder of message structure	
Overlarge/Zero/Negative value of scalar field	
Missing/multiple empty line in a message	
Mismatch between Method name and CSeq Method Name	
Presence of Request and Response line in a messages	
Missing Mandatory header fields in a Request Message	
Unknown scheme for “Request-URI”, header fields and message body	
Unknown Method Name and Response status	
Unknown scheme for message body	
Multiple Values in Single Value Required Optional Fields	
Tortured Messages	2,000,000
Overlarge Message Body	
Unknown header fields	
Extra white spaces between colons, semicolons, header field values	
Use of extra colons, semicolons in header fields	
Use of Escaped characters in messages	
Presence of empty optional header fields and parameters	
Line folding all over the message	
Presence of unknown and unusual header fields	
Integer fields (Max-Forwards and CSeq) with leading zeros	

The nature of the errors in these additional messages generated by “SIP-Msg-Gen” is described in Table 6.3.

The generation of “malicious” synthetic messages is aimed at stimulating an attack which will most certainly be characterized by a departure from a “normal” trace. Hence, we preliminary have performed a rough analysis to determine the characteristics of a “normal” trace, so that synthetic “bad” messages added to the trace can simulate an attack event or an intrusion attempt. To this end, we have examined the SIP traces collected over a period of eight months and split them among 24 time slots, one for each hour of the day. Features such as average arrival rate of call, register requests, traffic degradation ratio in the weekends, etc. are then extracted for each time slot. The values of these features are used to establish a base-line. A *message flooding attack* is simulated by instructing the “SIP-Msg-Gen” to generate a large number of SIP requests (here we use only INVITE and REGISTER request methods) with a rate that is much higher (two to three times) than the average rate in a normal situation. We have also created a few *legitimate traffic surge* scenarios which should not produce false positive classifications. Flooding attacks resembles legitimate traffic surges in the message arrival rate, but they differ in the fact that the legitimate high-rate INVITE tend to successfully establish a call through the standard three-way handshake INVITE/200 0K/ACK (Fig. 6.2 (a)) and terminate with a corresponding BYE message, while flooding attacks simply send a INVITE or REQUEST messages at high rate, without waiting for the full handshake to come through (Fig. 6.2 (b)).

Finally, *call tear down attack* scenarios are constructed by injecting a good amount of call requests (INVITE) messages and later those are rejected with CANCEL messages.

6.2.2 Kernel Selection and Parameter Estimation

Effective use of SVM requires an understanding of its parameters and their influence over classification accuracy [28]. In particular, the selection of a

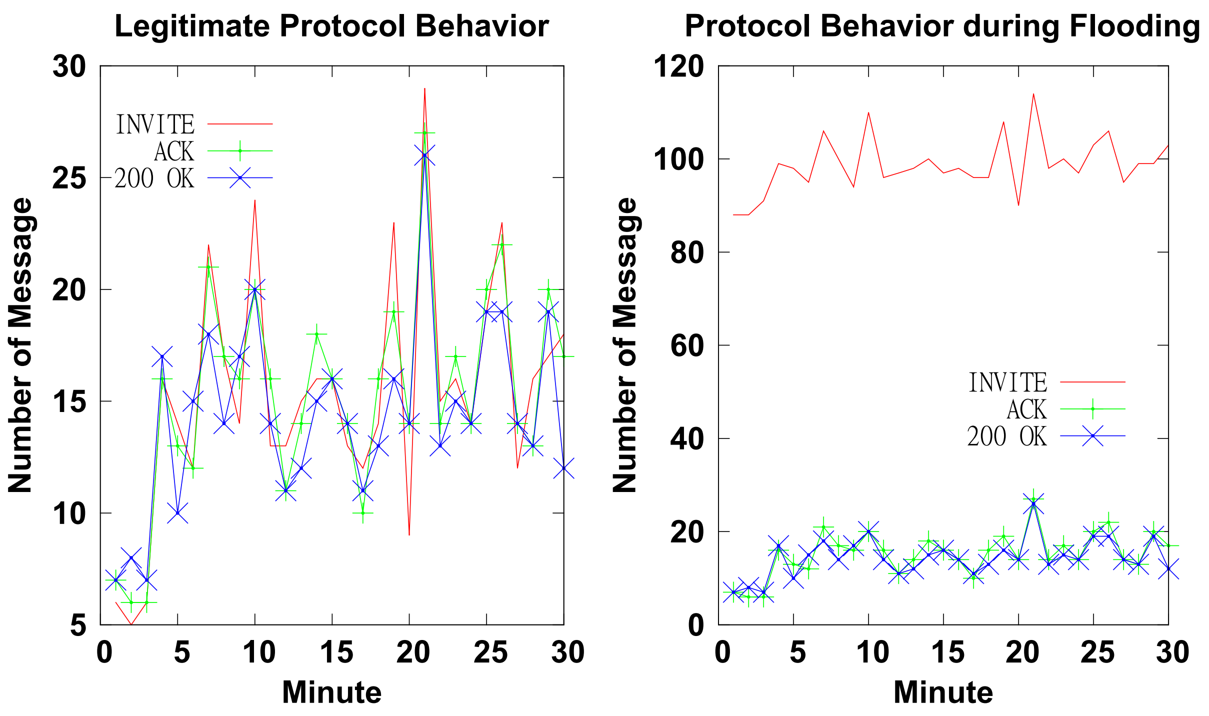


Figure 6.2: Behavior of SIP protocol attribute during (a) real legitimate normal traffic, and (b)INVITE flooding attack

Table 6.4: Description of synthetic "malicious" scenarios in the dataset

Scenario	Description	Duration	Test Cases	Total Messages
INVITE Flooding	Peak-hour in Weekdays	10 min	20	40,000
INVITE Flooding	Peak-hour in Weekdays	30 min	10	30,000
INVITE Flooding	In Weekends	10 min	10	1000
REGISTER Flooding	Peak-hour in Weekdays	30 min	10	20,000
REGISTER Flooding	Off Peak-hour in Weekdays	10 min	10	2,000
Call TearDown Attack	Peak-hour in Weekdays	10 min	10	5,000
Call TearDown Attack	Peak-hour in Weekdays	10 min	10	5,000

specific kernel function (such as linear, Radial Basis Functions (RBF), polynomial, *etc.*) and tuning of the kernel parameters (such as the RBF kernel parameter γ or the degree for polynomial kernel) can strongly influence the accuracy of the SVM classification. Another important parameter is the so called *soft margin constant* C which controls the trade-off between maximizing the margin and minimizing the training error.

There are no specific algorithms or techniques for selecting the most promising SVM configuration parameters, as this, much like in other machine learning techniques, turns out to be data dependent. However, there are several guidelines that help in parameter selection. As suggested in the support guide of LibSVM [87]) we made use of the widely used *cross-validation technique* [39] to estimate the probability of test error of a learning algorithm. In our search for a suitable parameter configuration, we performed a k-fold cross-validation. In our case we have used k=10 which means that the 800 SIP messages that were used for training of the SVM were randomly assigned to k=10 subsets. Each subset was then tested (validated) using the classifier trained on the remaining $(k-1)$ subsets using a specific set of parameters. The k results are averaged to obtain a single index. The entire process was repeated for each set of parameters. Finally, the parameters with the best cross-validation index were selected. This test was performed for three kernel functions: linear, RBF, and polynomial – and for various values of their relevant parameters, as shown in Table 6.5. The kernel function that is the easiest to tune is the linear kernel, as it has only the single C parameter to adjust. However, as expected, it did not perform well for this dataset. We then turned our attention to the RBF kernel that is often recommended despite its higher computational cost. Nevertheless, as shown in Table 6.5, its cross-validation accuracy with $0 < \sigma \leq 1$ is very low for small values of C (soft margin constant), although the accuracy increases as the value of C increases. However, small values of C are generally preferable since small values of C tend to emphasize the margin while ignoring the outliers in the training data and – conversely – large values of C increase the possibility of over fitting the training data. Overall, the RBF kernels were not satisfactory. The performance for various pairs of (C, d) values ($C = 2^{-1}, 2^0, 2^2, 2^3$... and $d = 2, 3, 4, ...$) were

Table 6.5: Cross-Validation accuracy of different kernel functions and parameters

Soft Margin Constant,C	Kernel Function	Accuracy
2^{-1}	Polynomial (degree 2)	99.23%
	Polynomial (degree 3)	97.56%
	Polynomial (degree 4)	96.27%
	Radial basis function (σ=0.4)	46.40%
	Linear	47.64%
2^{0}	Polynomial (degree 2)	99.96%
	Polynomial (degree 3)	97.56%
	Polynomial (degree 4)	96.50%
	Radial basis function (σ=0.4)	52.74%
	Linear	47.64%
2^{2}	Polynomial (degree 2)	99.23%
	Polynomial (degree 3)	97.16%
	Polynomial (degree 4)	96.59%
	Radial basis function (σ=0.4)	88.21%
	Linear	47.64%
2^{3}	Polynomial (degree 2)	99.23%
	Polynomial (degree 3)	97.02%
	Polynomial (degree 4)	96.65%
	Radial basis function (σ=0.4)	91.03%
	Linear	47.64%

evaluated through cross-validation and is shown in Table 6.5. The best combination was obtained for the pair C=1, d=2, indicating that a second degree polynomial is flexible enough to discriminate between the two classes with a reasonably small soft margin.

6.2.3 Performance Evaluation

The performance of the proposed architecture is measured through its *efficiency*, which is defined by the classification accuracy, and its *effectiveness*, which is the time/effort needed for classification.

The mixed dataset (combination of real and synthetic SIP messages described in 6.2.1) contains $252,817,093$ SIP messages. The achieved accuracy by the two classifiers is reported in the Table 6.6.

Table 6.6: SIP message classification result using SVMs

Description	Num. of Msg
Total Message	252,817,093
Real SIP messages ("good")	242,711,466
Real SIP messages with syntax error	2,627
Synthetic Message with syntax error	3,000,000
Synthetic (well-formed but not meaningful)	5,000,000
Synthetic (valid but "tortured")	2,000,000
Synthetic (well-formed and meaningful but malicious)	103,000 (80 scenarios)
Result of Lexical Analysis	
Discarded "malformed" Messages	3,002,627
Well-formed Messages (without syntax error)	249,814,466
First SVM Classifier Result(Detection of "crooked" messages)	
Number of Messages passed from lexical analyzer	249,814,466
True positive("bad" message identified as "bad")	4,976,016
False positive("good" message identified as "bad")	28,739
True negative("good" message identified as "good")	244,785,727
False negative("bad" message identified as "good")	23,984
Accuracy	99.97%
Second SVM Classifier Result (Detection of "malicious" scenarios)	
True positive("malicious" scenarios identified as "malicious")	78
False positive("good" scenarios identified as "malicious")	0
True negative("good" scenarios identified as "good")	0
False negative("malicious" scenarios identified as "good")	2
Accuracy	97%

All experiments are done in a machine of Intel Core i7 CPU, 2.0 GHz Quad-core and 8 GB RAM memory. The average time for the proposed filtering system to classify a SIP message is 0.50 millisecond/msg. This time is the aggregation of processing time of individual filtering stage.

It is found that about 0.40 millisecond/msg time is required by the lexical analyzer to perform syntax checking of a SIP message, while the two SVMs require 0.05 millisecond/msg each.

We have also looked at the latency between the onset of an attack and the time elapsed before the second SVM can detect it. In the test cases of attack scenarios (described in Table 6.4), the attack starts slowly from zero and increases until it reaches the maximum rate, then, the maximum rate is maintained constant until the attack stops. We noticed that the SVM classifier is able to detect the attacks as soon as the traffic reaches the maximum rate.

For example, among one of the attack test scenarios, excessive amount of call requests (INVITE) messages are injected into the good trace during 9.00-9.10 am in the morning on a Monday. The SVM classifier observes every SIP messages from 9.00 am and during 9.02 am it notices that the arrival of INVITE messages is much higher than the expected legitimate call request arrival rate at that specific hour of a week-day, percentages of the un-established and the percentage of the calls with very low interarrival time is high (than normal rate). These features indicate the occurrence of the *call flooding attack* and thus the classifier announces that the system is under attack.

Experiments with "inviteflood"

The performance of our proposed filtering system is also tested through "inviteflood"[1] which a publicly available security testing tool implemented by the author of [48] to perform SIP/SDP INVITE message flooding over UDP/IP. We use Asterisk IP PBX as the VoIP server as the attack tool is designed to use this. As the VoIP client, X-lite[2], an well-known soft phone application for desktop and mobile platforms, is used. To imitate the flooding attack scenario, we first create a small set of legitimate users by registering ten SIP accounts on the server. We have prepared a script that generates "good" calls among these user accounts for a specific duration of

[1] "inviteflood" - `http://www.hackingvoip.com/tools/inviteflood.tar.gz`

[2] X-Lite softphone :`http://www.counterpath.com/softphone-clients.html`

Table 6.7: Experiments performed on traces generated by "inviteflood" tool

Description	Value
Duration	4 hours
Total Call Requests	4,000
Number of "flooding" attack scenarios	10 (10 min each)
Second SVM Classifier Result	
True positive("malicious" scenarios identified as "malicious")	9
False positive("good" scenarios identified as "malicious")	0
True negative("good" scenarios identified as "good")	0
False negative("malicious" scenarios identified as "good")	1

time. On the other hand, using "inviteflood" tool a large number of call request are directed towards the set of "good" users by spoofing five SIP uris ("malicious" users). These calls are captured using the widely used open-source packet analyzer Wireshark[3]. Details about experiments with the tool "inviteflood" and performance accuracy is found in Table 6.7.

Experiments with PROTOS Test Suite

An experiments is performed using the publicly available PROTOS test suits to check the performance of our proposed filtering architecture. PROTOS test suite[4] is developed [201] for use in the evaluation of the implementation level security and robustness of SIP implementations.

PROTOS Test-suite:c07-sip contains 4527 test cases while the scope is only limited to the SIP INVITE messages. According to the given description of this dataset, 1 message is valid while the remaining 4526 messages are defined as *exceptional.* By the term *exceptional* the authors [201] mean messages that can violate the protocol specification, but often it is legal or in the hazy region between legal and illegal constructs. Among these *exceptional* messages, 193 messages are contain error in SIP method name (e.g., overflow-general, overflow-space, overflow-null, fmtstring, utf-

[3]Wireshark : `http://www.wireshark.org/`

[4]PROTOS Project, `https://www.ee.oulu.fi/research/ouspg/PROTOS_Test-Suite_c07-sip`

Table 6.8: SVM Classification result using PROTOS dataset

Description	Num. of Msg
True positive("bad" message identified as "bad")	4288
False positive("good" message identified as "bad")	0
True negative("good" message identified as "good")	239
False negative("bad" message identified as "good")	0

8, ansi-escape), 75 SIP messages contain error in SIP version description, 61 messages contain exceptional elements in SIP request URI, 2097 messages contain exceptional elements in various header fields, and finally 2100 messages contain error in the message body.

We performed an experiment with this PROTOS test suits and the classification result obtained from our proposed filtering system is found in Table 6.8.

6.2.4 Dimension Reduction

For each SIP message in the dataset (training and testing) a vector of 26 features was extracted. However, working with such a high-dimensional space is computationally heavy and perhaps unnecessary as some of the features thus extracted show to be correlated to one another. Hence, we used Principal Component Analysis (PCA) [94] to reduce the number of features used to represent data. The resulting dimensionality reduction would provide a simpler representation for the data, a reduction of the memory requirements and ultimately, a faster classification.

The result of applying PCA to our dataset is summarized in Fig. 6.3 which shows the "variance" and the "cumulative proportion of variance" of the derived principal components and from which we can see that the first four principal components that are maximally correlated already contain 92% of the information carried by the original 26 features and the first 17 principal components contain 100% of the information. This led to a third

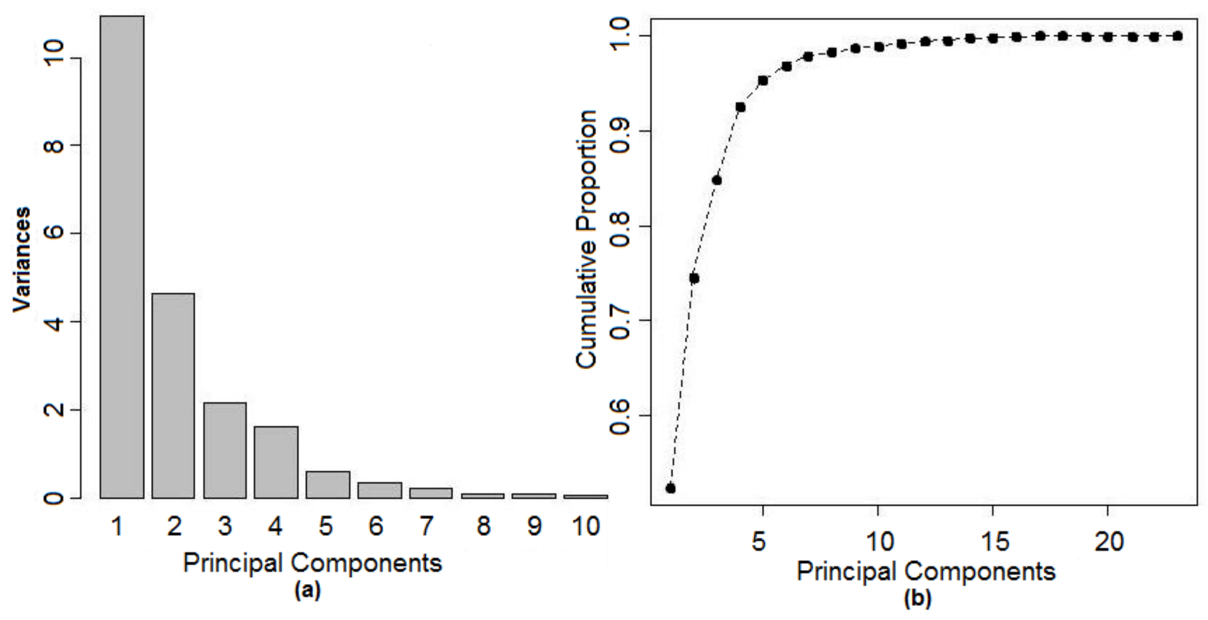


Figure 6.3: Results of Principal Component Analysis, (a) variance, and (b) cumulative proportion of principal components

experiment, in which the performance of the SVM classifier was evaluated when using only the first 17 principal components instead of the original 26 features. As expected, the accuracy of the classification using only the 17 principal components is almost as high as the one obtained using the original 26 features (99.45% v. 99.97%, respectively).

Chapter 7

Social Behavior Analysis of VoIP Users and its application to Malicious Users Detection

7.1 Description of the Dataset and Basic Characteristics

A sub-set of the collected SIP traffic from our institution is used for this analysis. The sub-set consists of SIP messages for a period of one and a half year (July 2012- December 2013). Around two million call attempts are retrieved from the captured 146 million SIP messages, among these one million calls are successfully established. For our analysis purpose each call is represented as a CDR (Call Detail Record) where each CDR consists of 5-tuples $\{x, y, t, l, n\}$, where user x is the caller, user y is the callee, t is the time of the call arrival, l is the duration of the call and n is the call status (e.g., successful or canceled). Fig. 7.1 shows the average call arrival rate per day in the dataset. Based on the arrival pattern of call request (INVITE message), it is noticed that the busy hour starts from around 8 am and ends around 5 pm. Fig. 7.2 shows the average interarrival time of call requests during weekdays and weekends. Fig. 7.3 shows that the

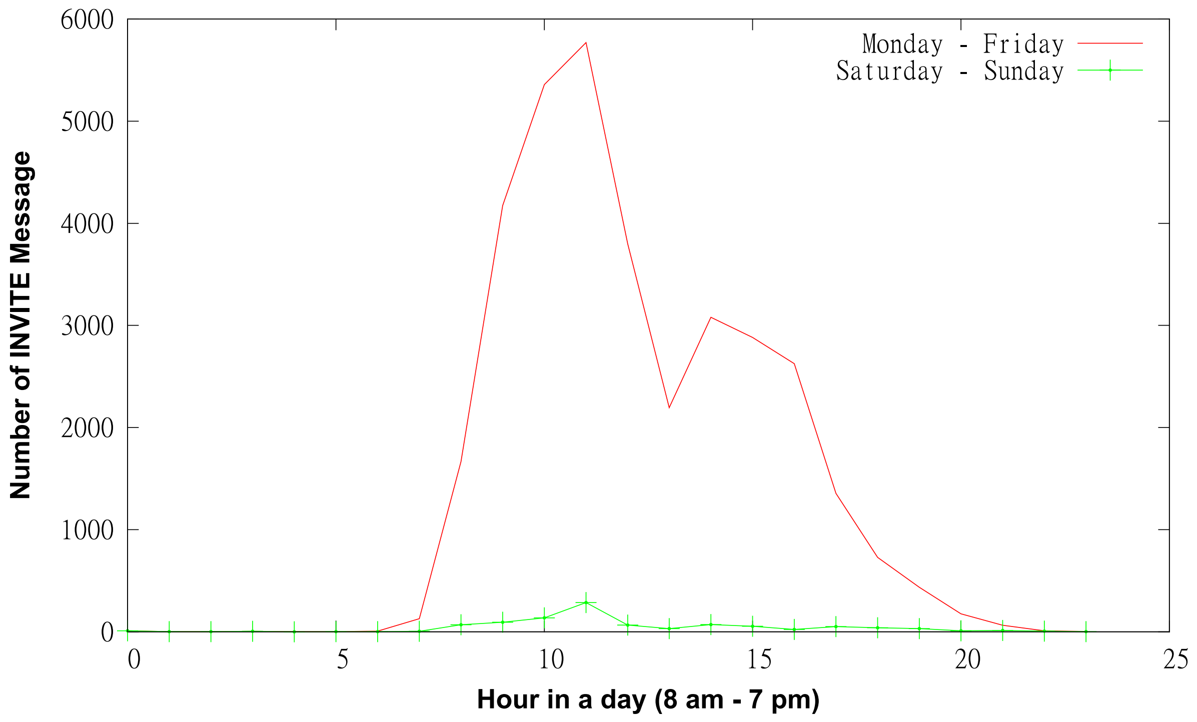


Figure 7.1: SIP INVITE Message arrival pattern in captured trace.

interarrival time of the call requests during the peak hours in weekdays follows exponential distribution.

Call Establishment Time

The average time for establishment of a call through a three-way handshake of INVITE/200 OK/ACK messages is shown in Fig. 7.4. Fig. 7.5 shows the distribution of the call establishment time during weekdays.

Call Duration Distribution

Fig. 7.6 shows the distribution of call durations during the observation period. For this plot, all the calls including unsuccessful ones are considered. That is why Fig. 7.6 shows a huge amount of calls with call duration less than 10 seconds which indicates the amount of rejected or canceled calls.

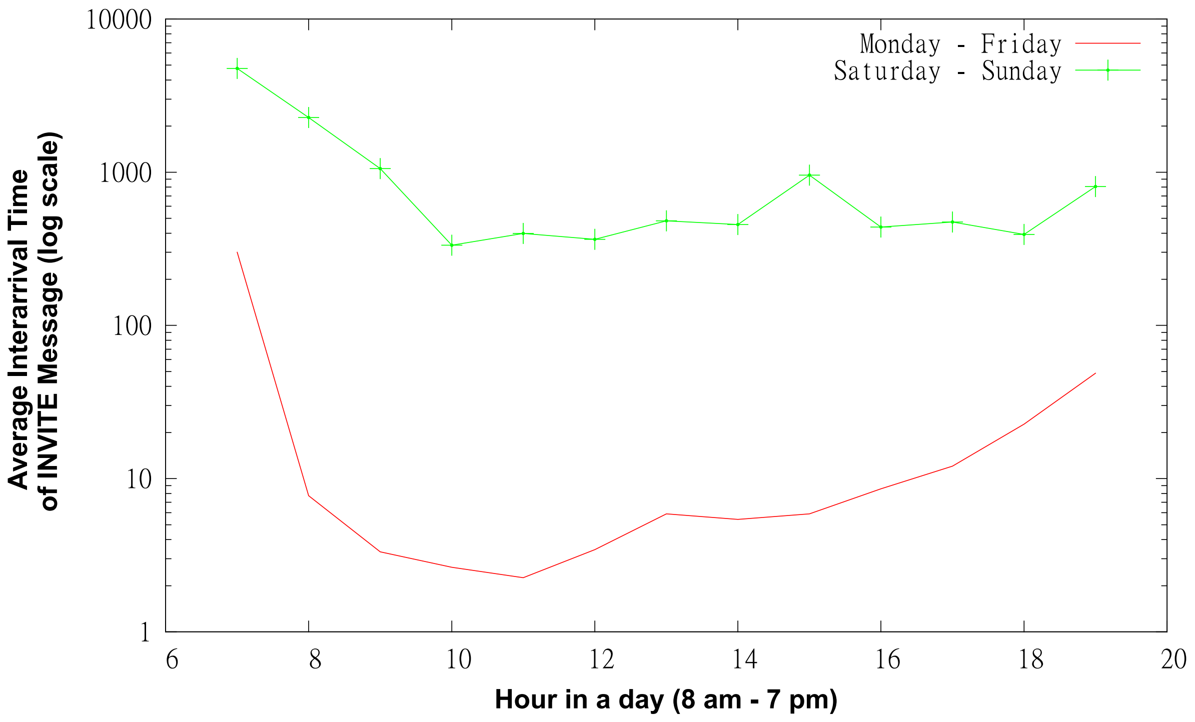


Figure 7.2: Average Interarrival time of SIP INVITE Messages in a day(8 am - 7 pm).

7.2 Social Behavior Analysis of VoIP Users

With the widespread adoption of SIP-based VoIP, understanding the characteristics of SIP traffic and the behavior of VoIP users is critical to problem diagnosis and security protection of IP Telephony. Information about the social behavioral patterns of the users is quite important in anomaly detection as these systems suspect potential intrusion by identifying deviations from the normal and legitimate behavior patterns of users and traffic.

To this end, we have analyzed the communication patterns of a large number of VoIP users with the aim to model the normal behavior of the users. A part of this work submitted for the peer-review is found in [55]. The behavior of the users can be modeled either on an individual or on a group basis, in such a way that the model captures the essence of the user's social behavior. Though it is quite common to analyze the individual

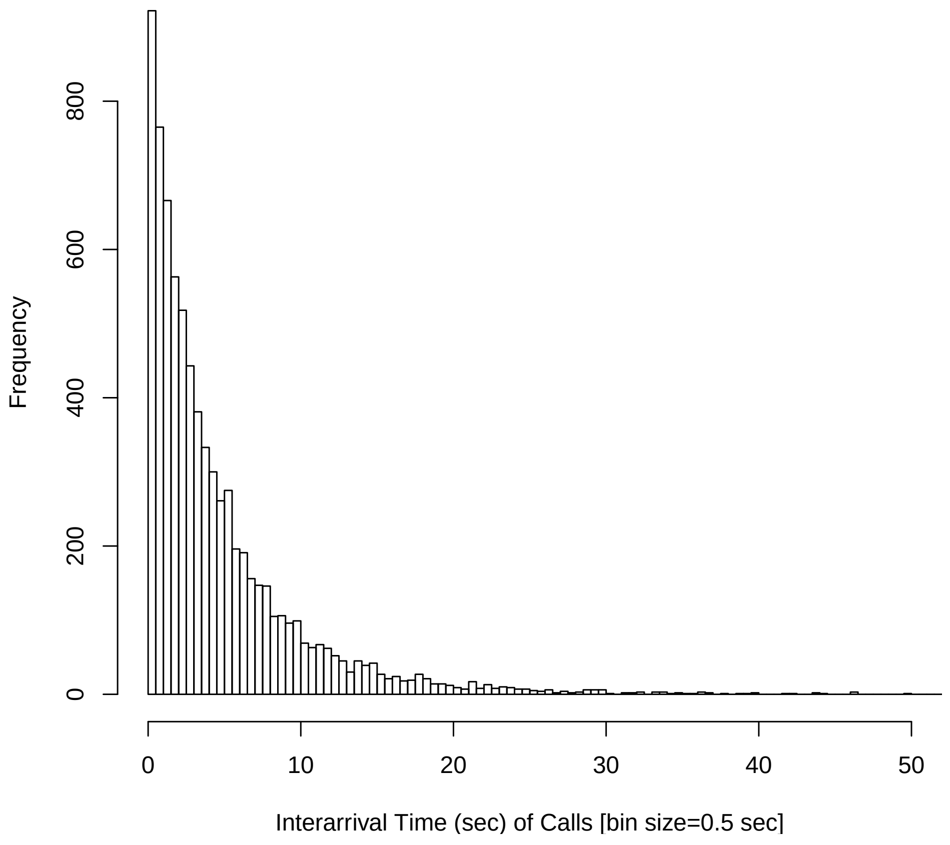


Figure 7.3: Distribution of Interarrival Time of Call Request in weekdays (8 am- 5pm).

attributes to understand user behavior (e.g., the average billing, frequency, amount and time of service usage, etc), following such brute-force approach to individually characterize millions of users is not a wise-decision in the context of telecommunication, where the network is huge.

7.2.1 Networks in Social Sciences

The concept of *network* in Social Sciences refers to the theoretical construct that is used to demonstrate the relationships between social entities such as, individuals, groups, organizations, etc. A *social network* is usually represented as a *graph* which comprised of a set of *nodes* or *vertices* connected by one or more *links* or *edges*. In this context, *nodes* are different social

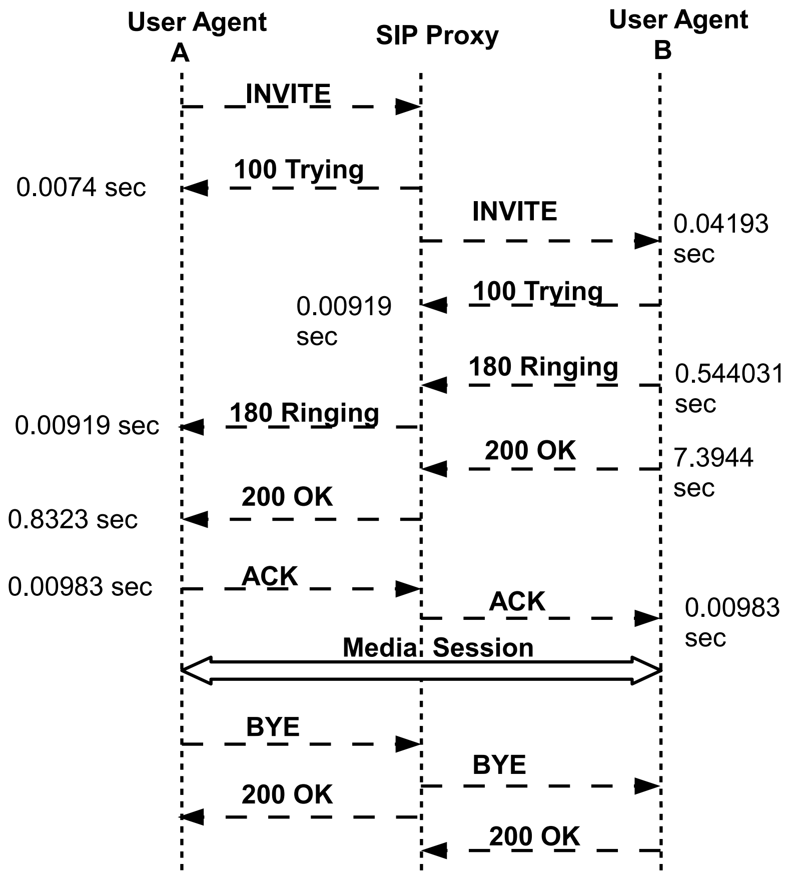


Figure 7.4: Computation of a call establishment time.

entities and the various relations which connect them are defined as *links*. A *social network* may be *undirected*, meaning that there is no distinction between the two *vertices* associated with each *edge*, or its *edges* may be directed from one *vertex* to the other. Various methods known as *social network* analysis (SNA) techniques [200] are applied on a social network to explore the social interaction pattern of users. For example, *degree*, *betweenness* and *closeness* centrality [19,21] are well-established and popular concepts to measure the social status and influence of a person inside a social network.

The notion of *social network* can be applied in the telecom networks too, where *social networks* of users are built based on their telephony records.

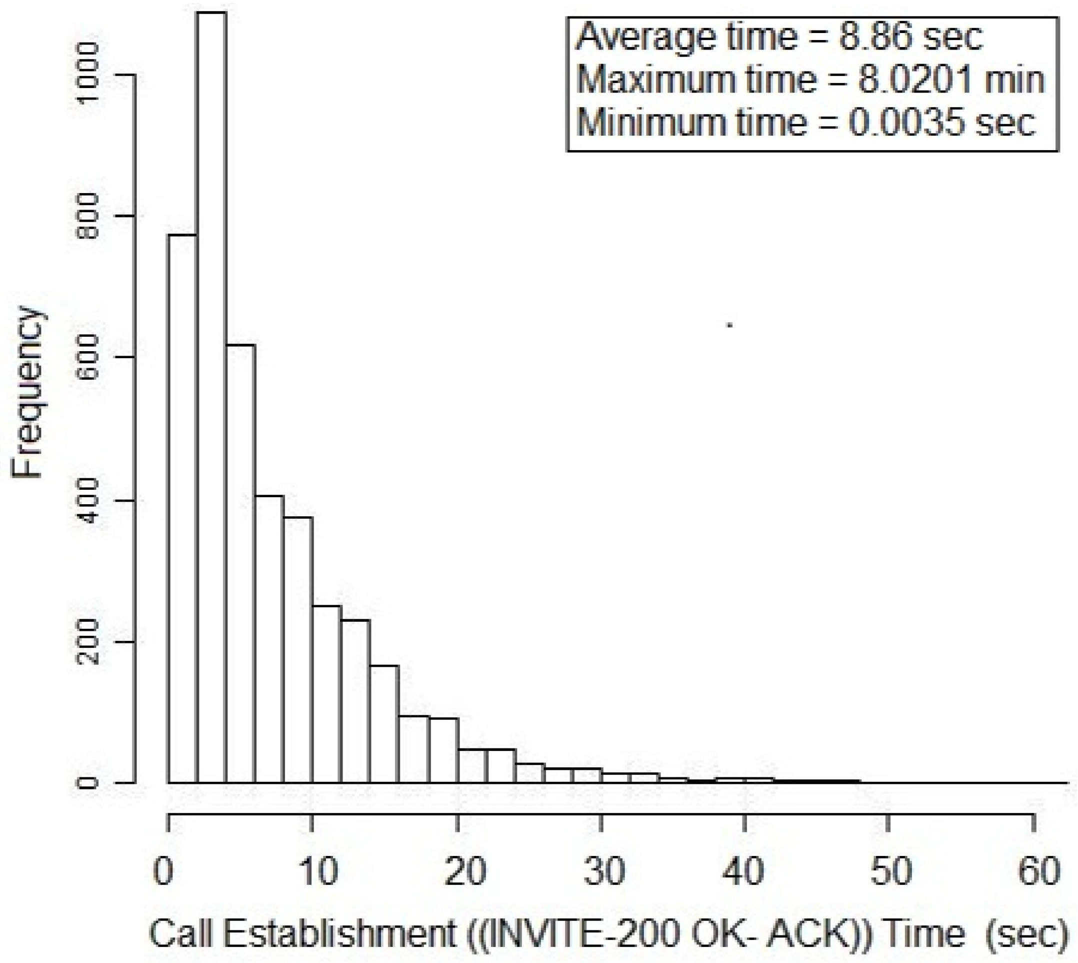


Figure 7.5: Distribution of Call Establishment Time.

Besides the individual user analysis, SNA techniques are very promising to analyze the complex behavior of telecom users [32, 135, 140, 174, 197] at different time-scales and at different levels of interpretation. In fact, application of SNA techniques in large amounts of telecom data is usually more relevant to reveal the behavioral patterns of users inside the network than the analysis of attributes of the individuals.

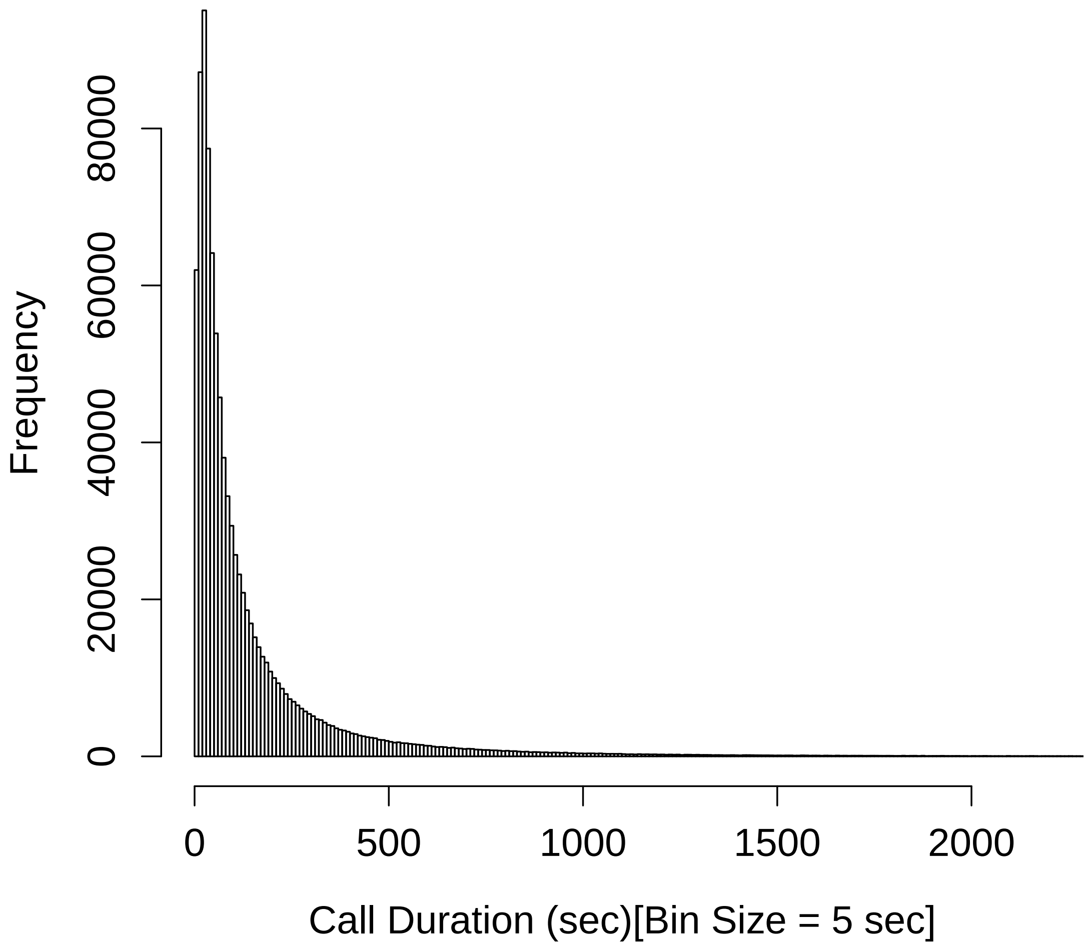


Figure 7.6: Distribution of Call Duration.

7.2.2 Social Networks of VoIP Users

After an initial analysis of the CDRs extracted from the captured SIP traces, a total of 126, 582 users[1] (SIP URIs) are found that appeared at least once during the whole observation window. In this context, we consider each SIP URI as an individual user. Of those, 8, 972 are found to be *internal users* and the remaining 117, 610 are identified as *external users.* The set of *internal users* consists of the SIP URIs of SIP end-points (e.g., softphones, IP phones, fax machines, etc) associated with administrative,

[1]We have excluded the SIP URIs from unknown sources.

Table 7.1: Number of users and percentage of calls generated by user groups

User Group	Num of Users	Call Percentage
Internal Active Users	2,551	68%
Internal Inactive Users	6,421	7%
External Active Users	475	6%
External Inactive Users	117,135	19%

technical, teaching and research stuffs, while, the set of *external users* contains SIP URIs of SIP end-points (e.g., IP phone, mobile, land line, fax machines, etc) outside the university domain. In this study, we construct a social network of these VoIP users from the data set consisting of their VoIP phone call records over a period of one year.

We have noticed that the *activity level* of a large number of users (both "internal" and "external") is very low. The *activity level* of a user is measured by the frequency of dialed and received calls by him/her. It is found that a large number of users appear only once or twice during the whole observation window to generate or receive calls. Thus, an *activity threshold* is used to differentiate the least active users from the most active ones. The threshold is defined based on the observed average activity level of users during the observation window. A user is defined as active if he/she remains active for at least 25% of the time of the observation period to generate or receive calls. This indicates that an active user dials or receives successful calls for at least 137 days (can be non-consecutive) during the observation window of total 548 days. The *activity threshold* is applied to both sets of internal and external users to divide them into groups of active, and inactive users. Table 7.1 shows the number of users and percentages of calls generated by the four user groups.

In our endeavor to construct a "social network" from the CDRs, we first concentrate on properly defining the set of "nodes" connected by one or more "links" (e.g., distinct types of relationships between "nodes"). In the context of VoIP, it is quite common that "nodes" are assigned to users and "links" are the communication (e.g., call, sms, video chat, etc) between them. A "link" in a social communication network can be characterized by several attributes such as, the frequency of communication, duration of communication, number of common neighbors, time since the last communication, etc. Each attribute has a different predictive power that also depends on the time scale at which we observe the "link".

We have only considered the set of "internal active" users (as "nodes") to build the social communication network. We have excluded the other groups of users shown in Table 7.1 because their activity is very limited during the observation window. It is found that a large number of users among $126,582$ users appear a very few times to make or receive calls during the one and a half year. Analyzing the behavior of a large number of *inactive* users does not provide any helpful indication in modeling the overall behavior and characteristics of the users of our considered telephony network. Moreover, if we consider the whole set of users to build the social network, then the network becomes huge, with a really large number of "nodes" (total $126,582$ users) and "links" (here, communication between users).

First, we build a caller-callee "social network" between the "internal active" users where the "link" between two "internal active" users is characterized by only the *frequency of communication* between them. This is a directed weighted call graph $G(V, E)$, where, $V(G)$ is the set of "vertices" representing the "internal active" users and $E(G)$ is the set of "edges" between them. An "edge" exists between two users if they have at least communicated once during the observation window and the weight of the "edge" is the frequency of communication between them. We measure the density of the call graph $G(V, E)$ to reveal how closely the group of "internal active" users are connected. Density of a graph is measured by the percentage of the number of actual ties/connection/links presents in the group relative to the number of possible links in the group (i.e., if everyone had a relationship with everyone else in the group). Density of the call graph $G(V, E)$ is 0.023 which indicates that the graph is very sparse (internal communication between the "internal active" users is not frequent). It is found that, this call graph captures only 10% information of the total calls generated during the observation window. Call graph $G(V, E)$ only explores the inter and intra departmental communication structure of the institute, it does not reveal the structure of social relationships of the "internal active" users with the outside world. This indicates that, a "link" between two users characterized by only the number of calls between them is not sufficient to capture the whole social relationship pattern of those users. Along with the *frequency of communication* between two users, few other features, such as the number of common neighbors, duration of communication can be helpful to predict the persistence of a "link" at a larger time scale.

Considering this, to explore the social connections between the group of "internal active" and other "internal" and "external" users , we have build another call graph $G1(V, E)$ by redefining the "links" between users. The call graph $G1(V, E)$ is an undirected weighted graph, where, $V(G1)$ is the set of "vertices" representing the "internal active" users and $E(G1)$ is the set of "edges" between them. Here, an "edge"/"link" between two users refers to the strength of social tie between them. The strength of the social tie between two users is expressed by a value in the range of 0 to 1. It is measured by the *similarity* of their call patterns (e.g., the number of common friends) and intensity of their internal communication (e.g., the frequency of communication).

- **Intensity of internal Communication**, $IC(i, j)$:
  This is measured by taking account the percentage of dialed calls between two users over the total number of calls generated by them.

For example, internal communication between users i and j is defined as follows:

$$IC(i,j) = \frac{\text{Num. of calls between user i and j}}{\text{Total num. of calls by i + Total num. of calls by j}}$$

- **Similarity of Call Pattern**, $J(i,j)$:
  Similarity of call pattern between two users is measured by considering the number of their common neighbors/friends. We measure it by using the Jaccard similarity coefficient[2] which a frequently used statistic measure for comparing the similarity and diversity of sample sets. It is defined as the size of the intersection divided by the size of the union of the sample sets. In this context, it is used to measure the percentage of common friends between two users. For example, the Jaccard similarity coefficient between user i and j is defined as follows:

  $$J(i,j) = \frac{|A \cap B|}{|A \cup B|}$$

  where A and B are the sets of users called by user i and by user j, respectively.

The social similarity/relationship measure, Sim(i,j) between two users i and j is then defined as the sum of the two indices above:

Sim(i,j)= J(i,j) + IC(i,j)

7.2.3 Community Detection

"Social networks" are built assuming that people are all interdependent inside a network. A typical observation is that, usually a member is linked to many other members, but not necessarily to all other member in the network. This introduces the concept of *community* structure, in which

[2] Jaccard, Paul. "The distribution of the flora in the alpine zone.1." New phytologist 11.2 (1912):37-50.

network "nodes" are joined together based on their behavior similarity in tightly-knit groups between which there are only looser connections. The concept of *community* is one of everyday familiarity and ubiquitous in society.

Community detection and analysis in a social network is quite relevant for revealing distinctive patterns of users inside networks. In telecommunication, for instance, where the entire network is huge and sparse, decomposition of the network into small communities (e.g., sets of highly interconnected nodes) is a promising approach to retrieve comprehensive information about the huge network. Identifications of smaller internal communities is helpful to discover complex social interaction pattern capturing highly connected circles of friends, families, or professional cliques in the observed network. Thus, the issue of detection and characterization of community structure in "social networks" has received a considerable amount of attention [16, 33, 69, 111, 130, 142]. We have performed different experiments to explore different community structures in the social network of the VoIP users (call graph $G1(V, E)$) that is represented in subsection 7.2.2. We want to discover structures in our *social communication* network rather than imposing a certain size of community or fix the number of communities. To this end, we go for unsupervised clustering to find out k communities. *K-means clustering algorithm* [78, 98], a widely used machine learning technique for unsupervised classification of data, is used here to divide the set of $2,551$ "internal active" users into pre-defined k communities/clusters[3] where users inside a community are of similar social behavior. Briefly, the k-means clustering algorithm takes a set of n observations and a (fixed) integer k $(1 <= k <= n)$ and proceeds to partition the n observations into k clusters so that observations belonging to

[3]We use the terms community and cluster as synonyms in this paper although in other contexts they may acquire slightly different meanings.

the same cluster to have a high “similarity” measure, while observations belonging to different clusters should have a very low “similarity” measure. Intuitively, observations in a cluster should be “densely packed”, while different clusters should be well “separated” from one another. Fig. 7.7 shows an example of clustering using k-means algorithm.

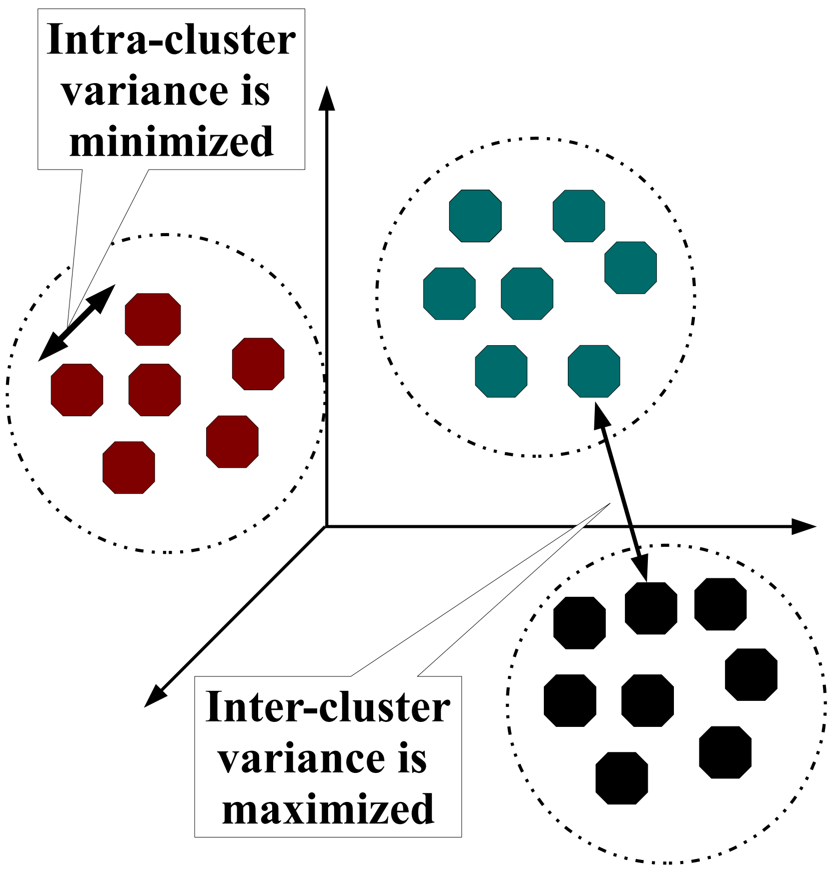

Intra-cluster variance is minimized
Inter-cluster variance is maximized

Figure 7.7: Example: k-means clustering.

Determining the number of Communities/Clusters (k)

Determining the optimal number of communities/clusters is essential for effective and efficient data clustering. The correct choice of k is often ambiguous and depends on the shape and scale of the distribution of points in a data set. There are several techniques of choosing the suitable number of clusters. Though neither of them guarantee to discover the "best" solution, but, can give an idea of the possible community structures.

We have started with the common method which is to compare the intra-cluster variance for a number of cluster solutions. It is noted that increasing k (number of cluster) without penalty will always reduce the amount of error in the resulting clustering process, to the extreme case of zero error if each data point is considered its own cluster (i.e., when k equals to the number of data points, n). Intuitively then, the optimal choice of k will strike a balance between maximum compression of the data using a single cluster, and maximum accuracy by assigning each data point to its own cluster.

The plot (in Fig. 7.8) of the intra-cluster variance ("within cluster sum of squares (WSS)") against a series of sequential clusters (here, 2<k<20) on our dataset provides a useful graphical way to choose an appropriate number of clusters with lowest intra-cluster distance. Intra-cluster distance of a cluster i is measured by the sum of squared Euclidean distances between the data instances and its center. It is known as "within cluster sum of squares (WSS)" and is calculated as :

$$\sum_{x_i^j \in S_i} \left(x_i^j - c_i\right)^2, \text{ where } c_i \text{ is the centroid of } i^{th} \text{ cluster} \tag{7.1}$$

Total "WSS" for k clusters can be evaluated as :

$$\sum_{i=1}^{k} \sum_{x_i^j \in S_i} \left(x_i^j - c_i\right)^2 \tag{7.2}$$

Usually for a clearly separable data set the reduction in "WSS" drops dramatically till a point with the increase of the number of cluster (k) and after that it reaches a plateau. This produces an "elbow" (The largest magnitude difference between two points) in the plot. The location of the "elbow" in the resulting plot indicates [103, 159, 186] that a satisfactory number of clusters have been reached. This “elbow” cannot always be unambiguously identified. For example, in case of our dataset we do not find such a sharp reduction (“elbow”) of “WSS” with the increase of the number of cluster (shown in Fig. 7.8). Thus, we cannot conclude any suitable solution that has a substantial impact on the clustering solutions.

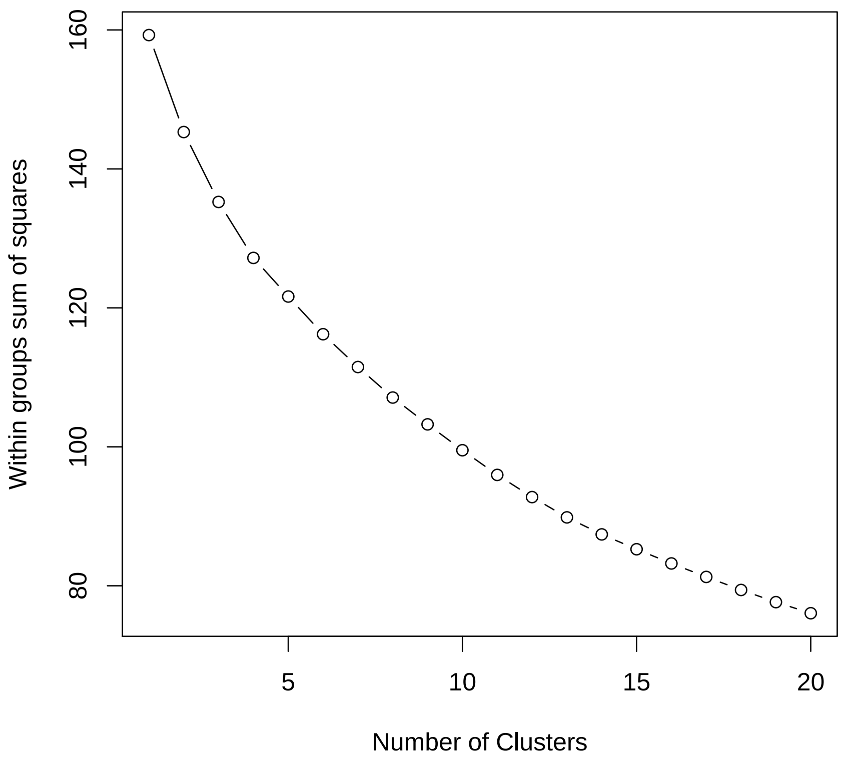


Figure 7.8: Intra-cluster distance for different values of k (number of clusters).

The modularity maximization approach is another widely used methods for community detection. Modularity is used to measure the strength of division of a network into communities. Networks with high modularity have dense connections between the nodes within communities but sparse connections between nodes in different communities. The modularity maximization method detects communities by searching over possible divisions of a network with high modularity. Here, we have used Louvain modularity method [16], a popular heuristic method based on modularity optimization algorithm, to discover communities in our *social communication* network of VoIP users. This community detection algorithm is applied on call graph $G1(V, E)$ (described in sub-section 7.2.2) in which "nodes" are the group of "internal active" users and "links" are characterized by the strength of social tie between them. At first, the community detection algorithm suggests a high number of communities for the call graph $G1(V, E)$ due to the reason that it is too sparse with a large number of "internal active" users with very low similarity. After filtering the "nodes" with similarity below 0.2, the community detection algorithm suggests that the remaining 87% of users can be divided into 15-20 communities.

We have also followed the expectation maximization in Gaussian Mixture Models to solve the problem of determining the optimum number of clusters [58]. Here, we assume that our dataset x is drawn from k Gaussian distributions,
$f(x) = \sum_{i=1}^{k} \lambda_i f_i(x)$,
where f_i is the probability density function of the observations in group k, and λ_i is the probability that an observation comes from the k^{th} mixture component ($\lambda_i \in (0, 1)$ and $\sum_{i=1}^{k} \lambda_i = 1$). The Expectation-Maximization algorithm is used to estimate the parameter (number of Gaussian distributions in the mixture density/number of cluster) through maximum likelihood. In selecting the "best" model that fits our data, we extend the use of Expectation-Maximization algorithm within a varying range of possible number of clusters (1<k<20) and then models are compared using Bayesian Information Criterion (BIC) [162], a statistical criterion for model selection. Fig. 7.9 shows the result of such model based approach. Model with the highest BIC indicates the "best" model to fit the data. Fig. 7.9 shows that the best model is "VEV" which indicates that the model is ellipsoidal with 15/16 components/clusters where the shapes of all clusters are equal while their volume and orientation may vary. Details about all

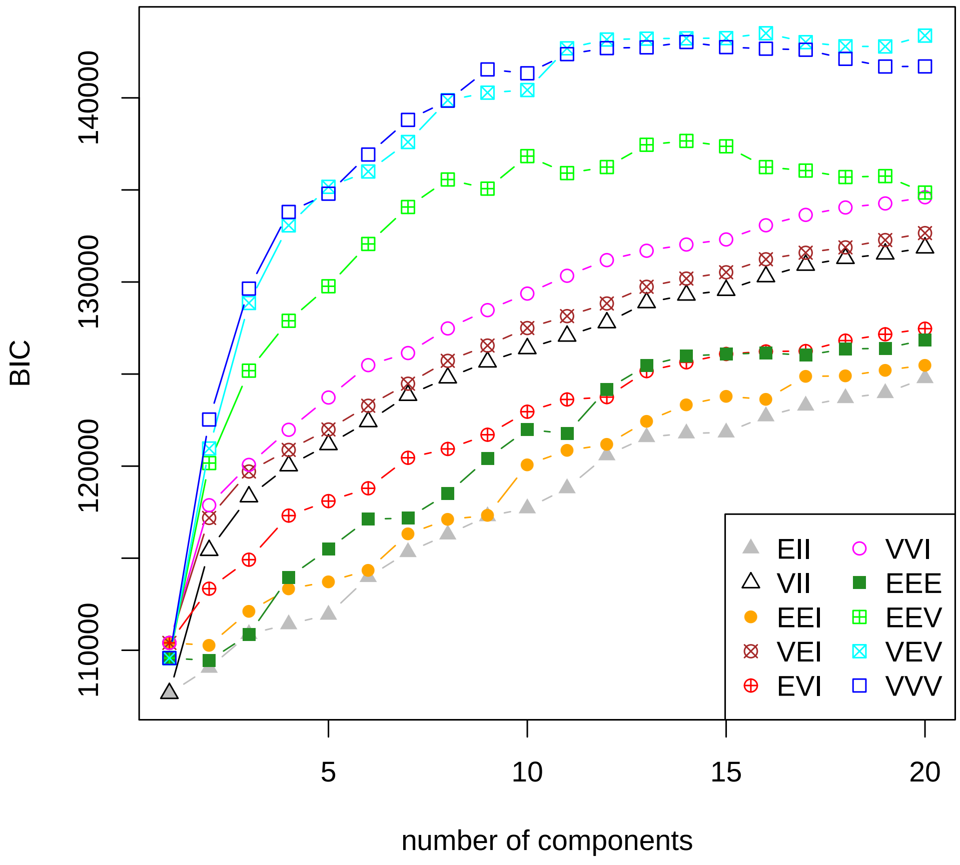


Figure 7.9: Selection of best model using Bayesian information criterion (BIC).

the models is found in [59]. Based on the above experiments we decide to choose k= 15 as the suitable number of communities/clusters to divide our social network.

7.2.4 Community Analysis & Profile Extraction

By applying the k-means clustering algorithm the set of "internal active" users are divided into 15 communities/clusters. Fig. 7.10 shows the 15 clusters of users in two dimensional space. In the figure the clusters are

Table 7.2: Importance of 12 components

Comp. No	1	2	3	4	5	6	7	8	9	10	11	12
Standard deviation	0.09	0.08	0.07	0.07	0.06	0.06	0.06	0.06	0.06	0.05	0.05	0.05
Proportion of Variance	0.15	0.13	0.11	0.10	0.08	0.08	0.07	0.06	0.06	0.06	0.05	0.05
Cumulative Proportion	0.15	0.28	0.39	0.49	0.57	0.65	0.72	0.78	0.84	0.90	0.95	1.00

not clearly distinguishable as the first two components cover only 28% of the total variance (shown in Table 7.2) among the 12 components/features that are used to describe our dataset.

Fig. 7.11 shows the call flow between clusters of “internal active” users, and the remaining users groups (e.g., “internal inactive”, “external active” and “external inactive”).

We perform a deep analysis of these observed communities/clusters of users to discover the patters of social relationships inside each community. This analysis helps us to explore and model the normal and *legitimate* behavioral profile of the “internal” users. This study of communities, whether densely or sparsely connected, is quite relevant to apply distinct security solutions for users with different behavioral profiles.

A list of features used to describe the social behavioral patterns of users is described below. The following two metrics are used to describe the structure and cohesion of a community/cluster:

- *Network Density*–Density captures how closely a community is connected. It is measured by the percentage of the number of actual ties/connection/links presents in the group relative to the number of possible links in the group (i .e., if everyone has a relationship with everyone else in the group).

- *Average path length*–It is the measure of the average length that one requires to reach any node from any other node in the network.

The following three features are used to explore the interactivity and

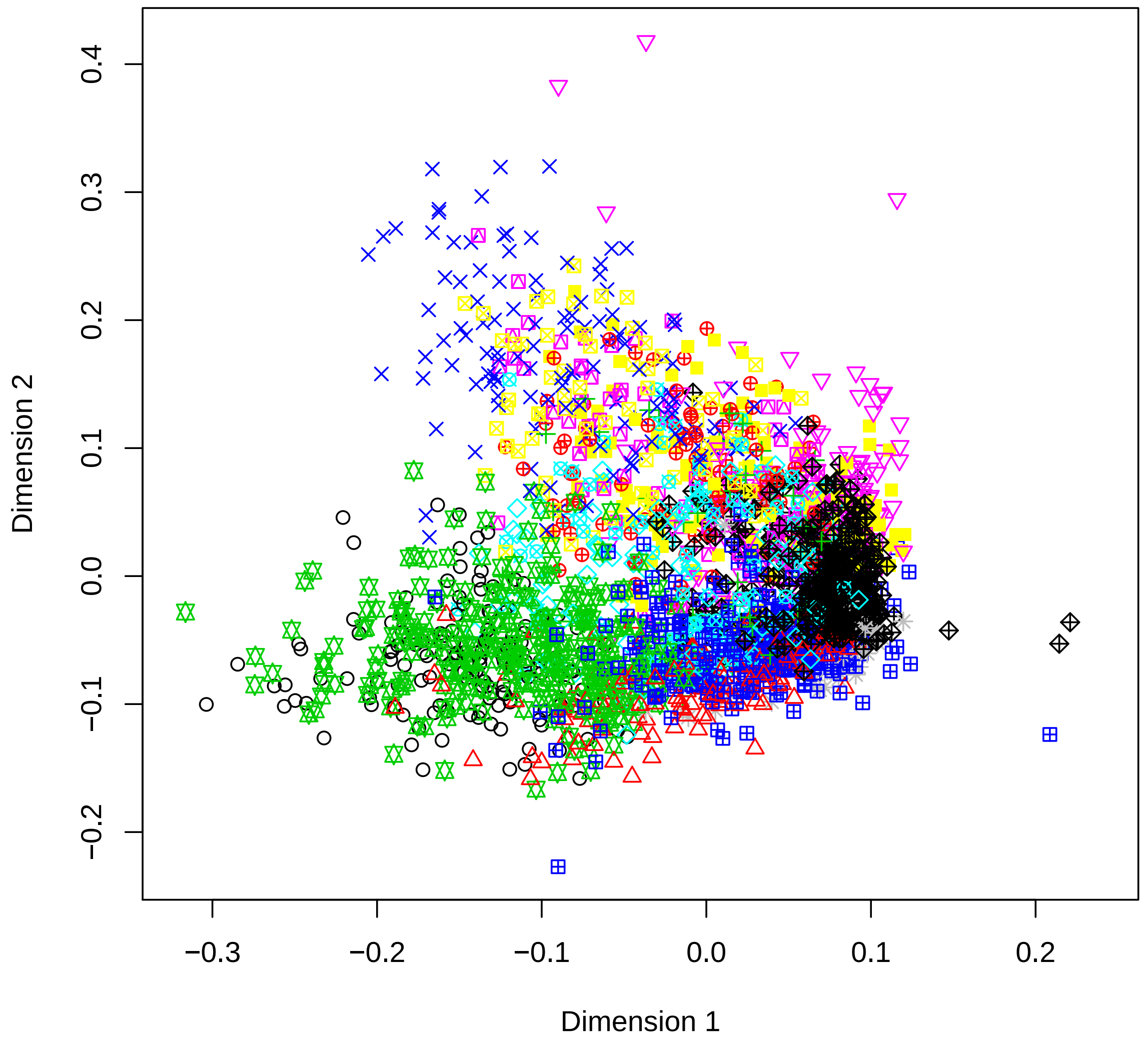


Figure 7.10: Clusters of users based on social behavior similarity. Clusters are not clearly separable using only two components.

dynamicity of users inside a community:

- **Interactivity of User**–Defined by the ratio of the incoming and outgoing calls (of a user) during the observation period.
  Interactivity of User A $= \frac{\text{Dialed Call by User A}}{\text{Received Call by User A}}$

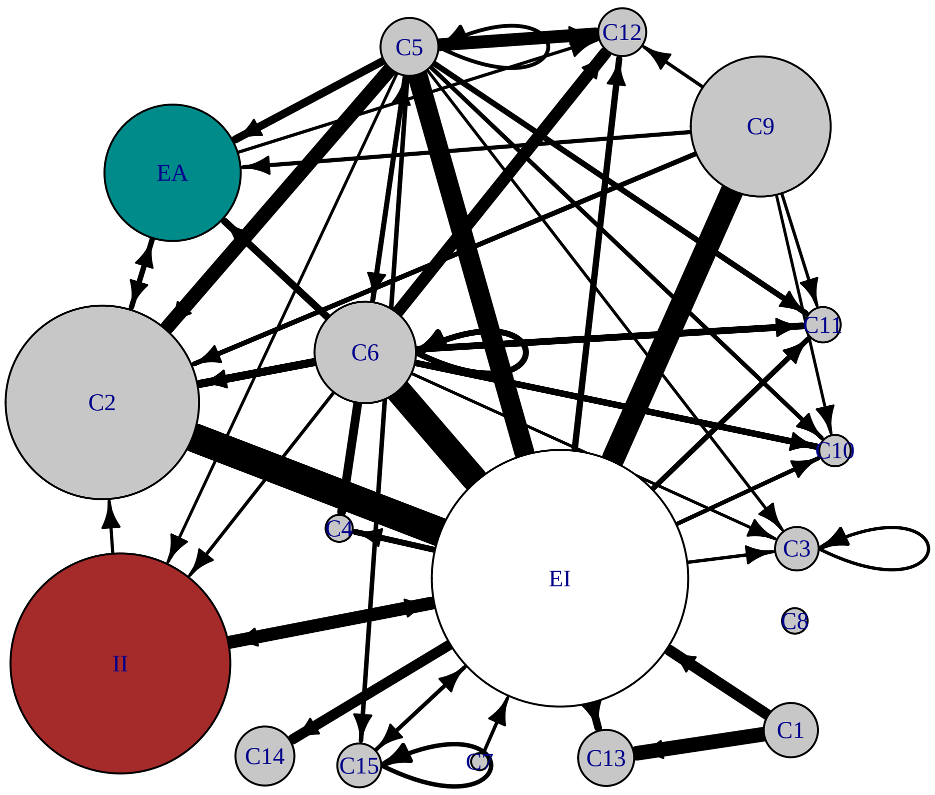


Figure 7.11: Communication between user groups. Vertex label C1-C15 indicates clusters of "internal active" users, II, EA and EI indicates "internal inactive", "external active" and "external inactive" users. Size of the vertex corresponds to the size of the user group, smallest one indicates at least 50 users. The thinnest edge between two groups indicates at least 4000 calls.)

- **Activity Duration of User**–Defined by the ratio of the time a user remains active and the total observation window.
  Activity Duration of user A $= \frac{\text{Period when user A remain active}}{\text{Total observation period}}$

- **Dynamicity & Sociality of Users**–A user is called "social" if he intend to call to a variety of people instead of maintaining communication only with fixed set of people. These is measured through the features "sociality" and "dynamicity":
  Sociality of User A $= \frac{\text{Number of contacted Callees by User A}}{\text{Total calls generated by User A}}$

Dynamicity of User A $= \frac{\text{Number of contacted Callees by User A}}{\text{Total Number of Users}}$
In our analysis, a users can be tagged as "social" if it is found that during the observation window his "sociality">=0.4 and "dynamicity">0.06.

The following metrics are used to identify the relative power, influence and importance of a user inside a community:

- **Degree Centrality**–In a "social network" the degree centrality is defined as the number of ties that a node has. In our context, we define the degree of a user as the total number of people he/she reaches directly (dials and receives calls) during the observation window. There are usually two separate measures of *degree* centrality, namely *in-degree* and *out-degree.* In telephone communication scenario, *in-degree* of a user refers to the number of calls he/she receives, while, *out-degree* refers to the number of calls he/she dials during the observation window.

- **Betweenness Centrality**–In Social Sciences, *betweenness centrality* [21] is a measure of a node's centrality in a "social network". It is equal to the number of shortest paths from all vertices to all others that pass through that node. Betweenness centrality is a more useful measure (than just connectivity) of both the load and importance of a node. The higher this parameter, the more influential the node is. The nodes which have high betweenness centrality are not necessarily the ones that have the most connections and do not have to be the most popular ones. In our context, it measures how likely a person falls in the most direct route between two people in the *social communication* network.

- **Closeness Centrality**–The average distance from a given node to all other nodes in the network. In our context, it is the measure of how fast a person can reach everyone else in the network.

The following concepts are used in describing the social behavior of users:

- **Neighbor**–Neighbors of a cluster i indicate the set of users (from other user groups) who are communicated by the members of cluster i during the observation window. It is defined as N_i.
- **Communication Network**–In order to map and measure the relationship and communication flows between members of a cluster and users from other groups, we define the concept of *community network.* The *communication network* for a cluster consists of its members and neighbors and defined as CN_i, where i = Number of clusters. It is defined as:
  M_i = Number of Users inside $Cluster_i$
  N_i = Number of Users Communicated by members of $Cluster_i$
  $CN_i = M_i + N_i$
- **Sub-Network**–A *sub-network* refers to the strongly connected small community/group inside a *communication network.* It is defined as SN_i^j, where i = Number of clusters, and j = Index of sub-network.

After analyzing the connection and interaction of users of 15 clusters, five different social behavioral patterns are noticed. The five social behavioral patterns of the "internal" users are described in Table 7.3 using a set of features. A brief description of these patterns is found below.

User Behavior Pattern 1: Medium Volume Caller

Users of three different communities/clusters ($cluster_1$, $cluster_7$ and $cluster_9$) follow this behavioral pattern. The main characteristic of these users is that they generate at least three times higher than the number of calls they receive during the observation window. Though these users usually

Table 7.3: Five distinct social behavioral patterns of "internal active" users

Feature	Pattern 1	Pattern 2	Pattern 3	Pattern 4	Pattern 5
Description	*Medium Caller*	*Heavy Caller*	*Medium Receiver*	*Heavy Receiver*	*Both Caller & Receiver*
Num. of users	577	435	161	541	837
Num. of clusters	3	2	1	5	4
Num. of "social" user	38	63	24	52	40
Activity Duration	60%	80%	80%	68%	80%
Dialed Call Percentage	25%	53%	0.84%	0.67%	20.49%
Received Call Percentage	3.52%	12.80%	5.89%	44.91%	32.88%
Active Hour in a day	8-17	8-17	8-17	8-17	8-17
Inactive Months	August & December	August & December	August & December	August & December	August & December
Average Call Duration	4 min	4 min	4 min	4 min	4 min

generate more calls than they receive, the volume of the outgoing calls is not too high. On average they generate less than 400 calls per year. We define users of these communities as *medium volume callers*.

Users of these 3 clusters generate 25% calls of the total calls generated by all the clusters of "internal active" users. Though, here users generate huge call, most of their calls are directed to other groups of "internal" and "external" users. That is why, users of the same clusters are very lightly connected with each other. This indicates that users of these three clusters are grouped together based on their similarity of common friends outside their clusters instead of direct communication between them. For instance, Fig. 7.12 shows the internal communication between users of $cluster_1$. It shows that, more that 90% of the users of $cluster_1$ has communicated with each other for less than 10 times in a year. Most of the users of these clusters remain active at least for 60% time of the observation window. The average duration of the dialed and received calls by users of these clusters is 4 minutes. Most of the calls generated and received by users of these clusters are during peak hour of working days.

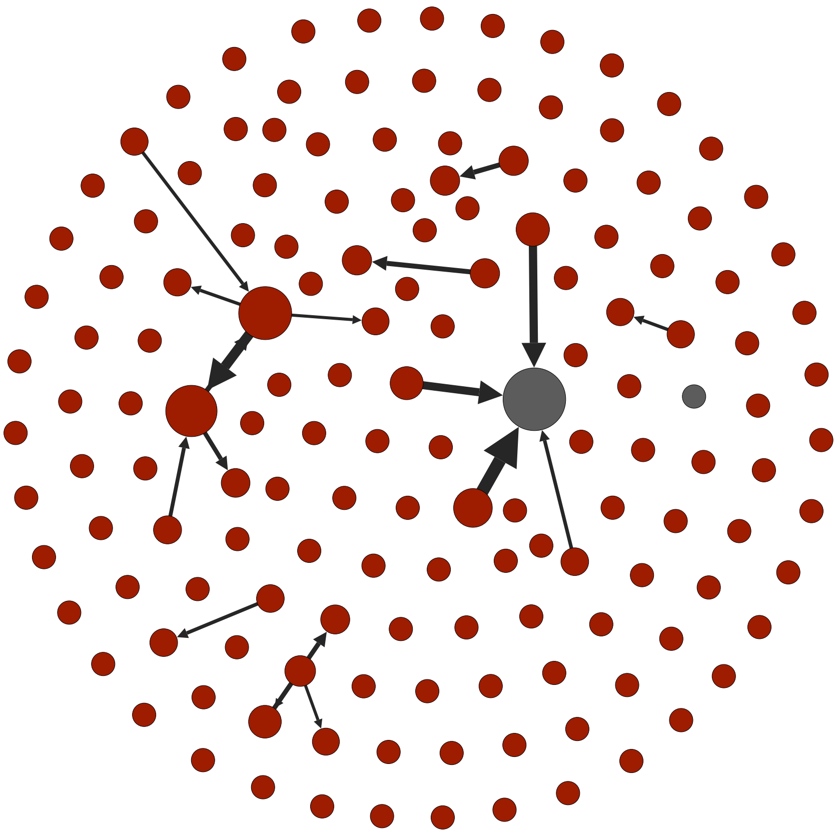


Figure 7.12: Internal communication of users inside $cluster_1$. The thinnest edge indicates 10 calls. Red filled "circles" indicate callers and grey filled "circles" indicate receivers. Size of the *vertex* corresponds to the frequency of dialed & received called by user.

We also consider how users of these three clusters communicate with other users (both "internal" and "external"). To this end, we consider the *communication network* for each three clusters. For instance, the *communication network* of $cluster_1$, CN_1 consists of 8219 users (148 members of $cluster_1$ and 8205 neighbors who are communicated by members of $cluster_1$ during the observation window). We find out that CN_1 is very sparse. About 70% of the neighbors are "external" users and a large number of them appear very few times during the observation window. Only 76 users out of 8219 are found who appear more than 100 times in a year to communicate other users of CN_1. Fig. 7.13 shows the interaction pattern of these users in CN_1. It is noticed that, among these 76 users only 6 are from $cluster_1$, while the rest of them are from other “internal” and “external” groups. Thus, we can conclude that Fig. 7.13 shows the social interaction of the most active 6 members of $cluster_1$ where they maintain a strong communication relation with other 70 users from the groups of “internal” and “external” users. It is found that, the these active 72 users in CN_1 are not fully connected, considering their call history for the observation window, these users are divided into 7 sub-networks. Fig. 7.14 shows the sub-networks of these users inside CN_1 and also the betweenness centrality of each users.

User Behavior Pattern 2: High Volume Caller

Users of two different communities ($cluster_5$ and $cluster_6$) follow this behavioral patter. The main characteristic of the communication pattern of these users is that they generate at least six times higher number of calls than they receive during the observation window. Users of these two clusters generate 53% calls of the total calls generated by all the clusters of the “internal active” users. Due to the high volume of outgoing calls generated by users of these communities (on average they generate more than 600 calls per year), they are defined as *high volume caller*.

70% of the calls generated by these users are towards other “internal” and “external” groups of users, while, the remaining 30% of their calls are generated and received by users of the same cluster. That is why, uses inside these two communities are lightly connected. This indicates that users of these communities are grouped together based on both their similarity of common friends with whom they communicated during the observation period and their frequency of internal communication. For instance, Fig. 7.15 shows the 30% of the users of $cluster_5$ that has communicated with each other at least 30 times during a year.

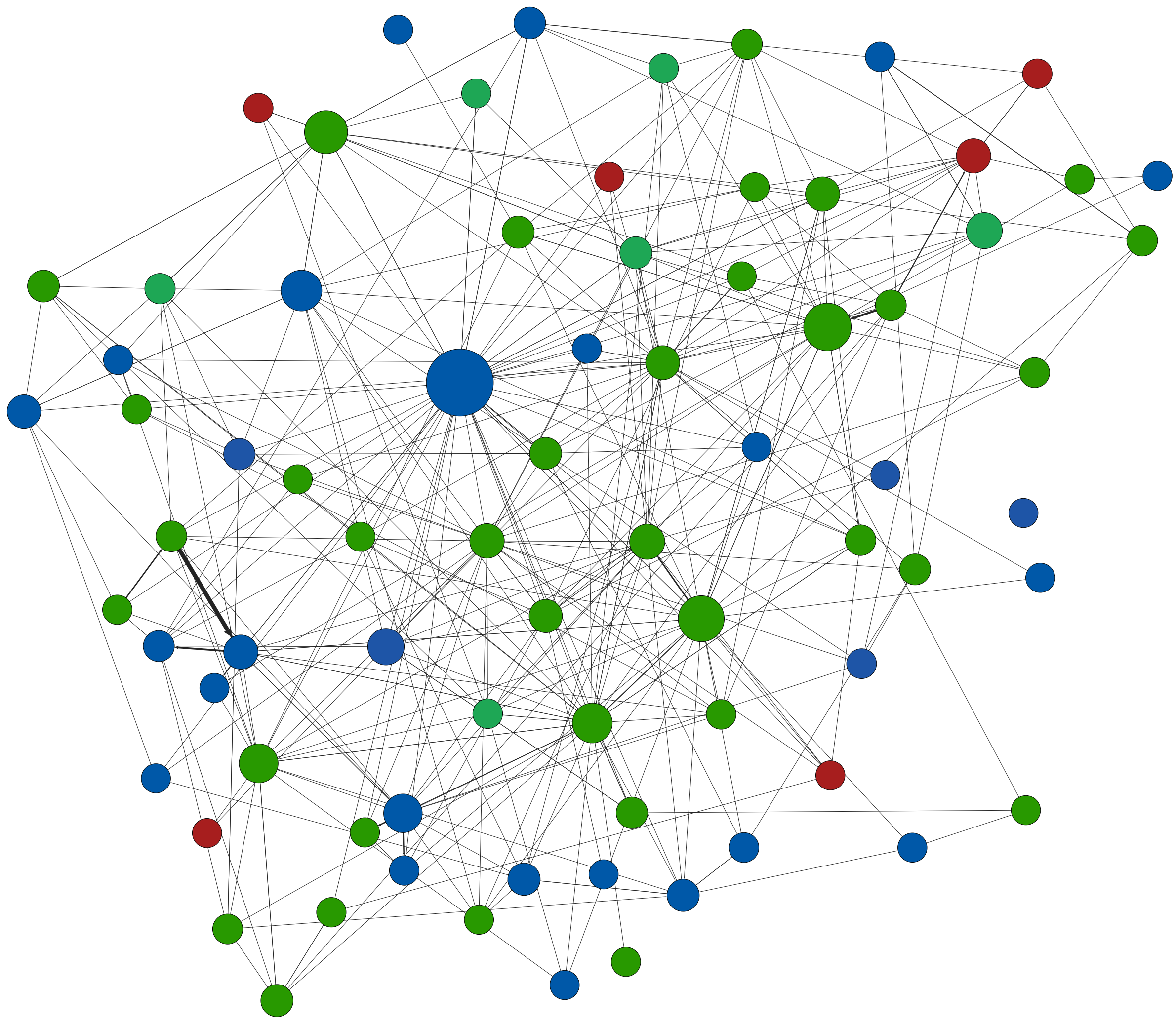


Figure 7.13: *Communication network* of *cluster*$_1$, CN_1. The thinnest edge indicates 100 calls and size of the *vertex* corresponds to the frequency of dialed & received call by each user. Red "circles" indicate members of *cluster*$_1$, green "circles" indicate *internal* users, and, blue "circles" indicate *external* users.

Most of the users of these clusters remain active at least for 80% time of the observation window. The average duration of the dialed and received calls by users of these clusters is 4 minutes. We also consider how users of these three clusters communicate with other users (both "internal" and "external"). To this end, we consider the *communication network* for each

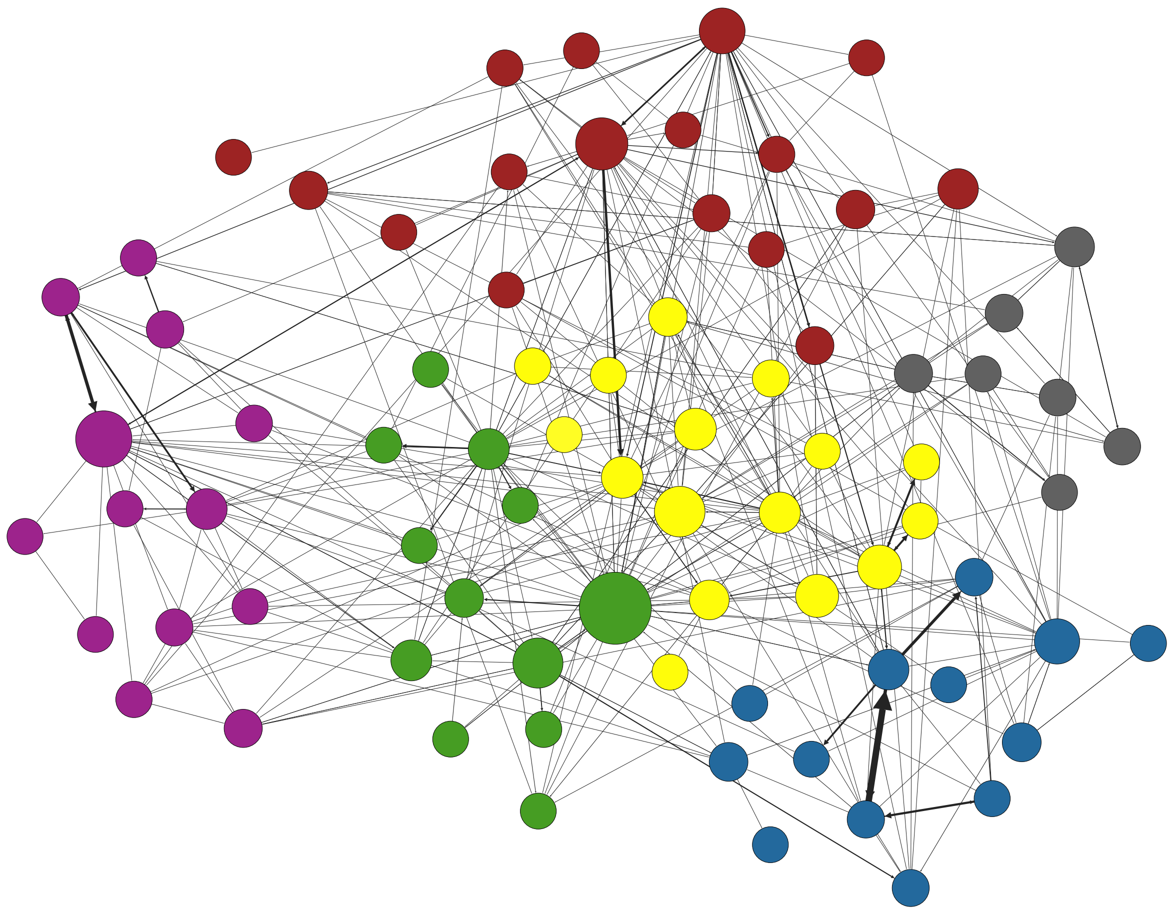


Figure 7.14: *Sub-networks* inside the *Communication network* of $cluster_1$. The thinnest edge indicates 100 calls and size of the *vertex* corresponds to *betweenness centrality* of each user. *Vertices* with the same color belong to the same *sub-networks*.

three clusters which consists of members and their neighbors. For instance, the *communication network* of $cluster_5$, CN_5 consists of 20579 users (158 members of this cluster and 20421 neighbors who are communicated with members of $cluster_5$ during the observation window). At first glance, we find this communication network to be very sparse due to the fact that 80% of the communicated users are "external". That is why, around 60% members of $cluster_5$ appears to be very *social* (e.g., these users tend to dial and receive calls from a large variety of people). Later, it is noticed that

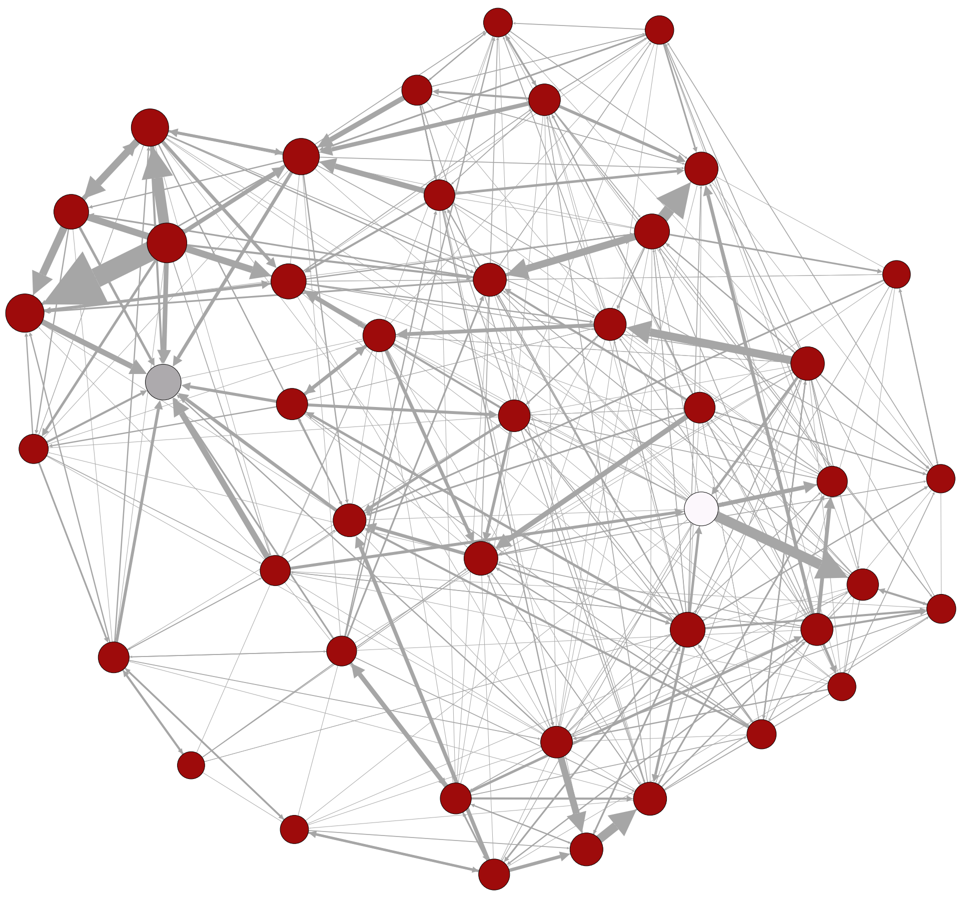


Figure 7.15: Internal communication of users inside $cluster_5$. The thinnest edge indicates 30 calls. Red filled "circles" indicate callers, grey filled "circle" indicates receivers, and, white "circle" indicates both caller and receiver. Size of the *vertex* corresponds to the frequency of dialed & received called by user.

most of these "external" users appear very few times during the observation window. If we exclude the users that generate or receive less than 350 calls per year from CN_5, then we find a well-connected network which consists of only 52 of 20579 users. Figure 7.16 shows the social interaction of these 52 users in CN_5 and it is noticed that all of them are members of $cluster_5$. Based on their call pattern, these users are further divided into 5 strong sub-networks. This indicates that though each user of $cluster_5$ communicated with a large number of users (both "internal" and "external") of

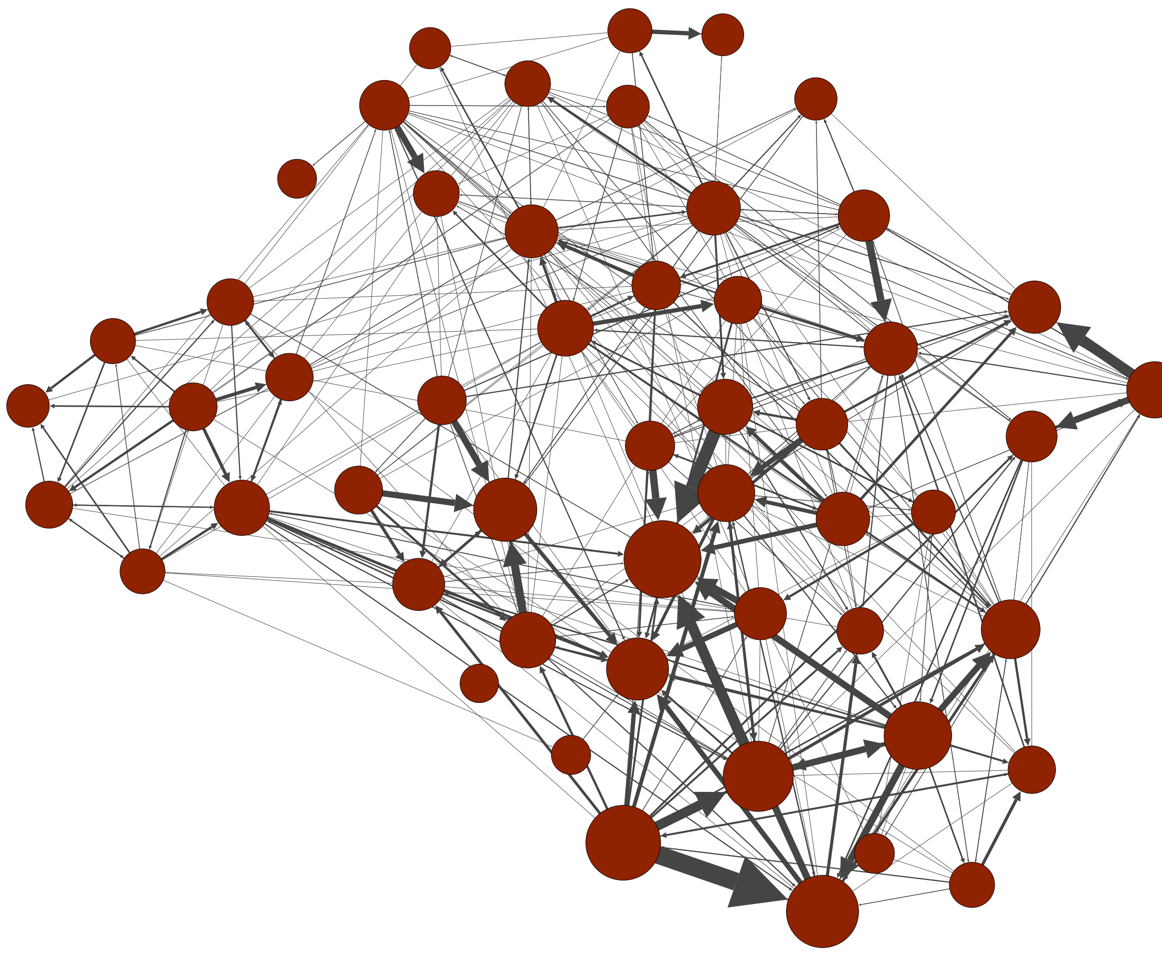


Figure 7.16: *Communication network* of $cluster_5$, CN_5 (Thinnest edge indicates 350 call, size of the *vertex* corresponds to the frequency of dialed & received call by each user). Red "circles" indicate members of $cluster_5$.

outside their cluster, in the end, the most active and well-connected users maintain a regular communication with a small number of people within the same cluster. Fig. 7.17 shows the sub-networks of 52 users inside CN_5 and also the betweenness centrality of each users.

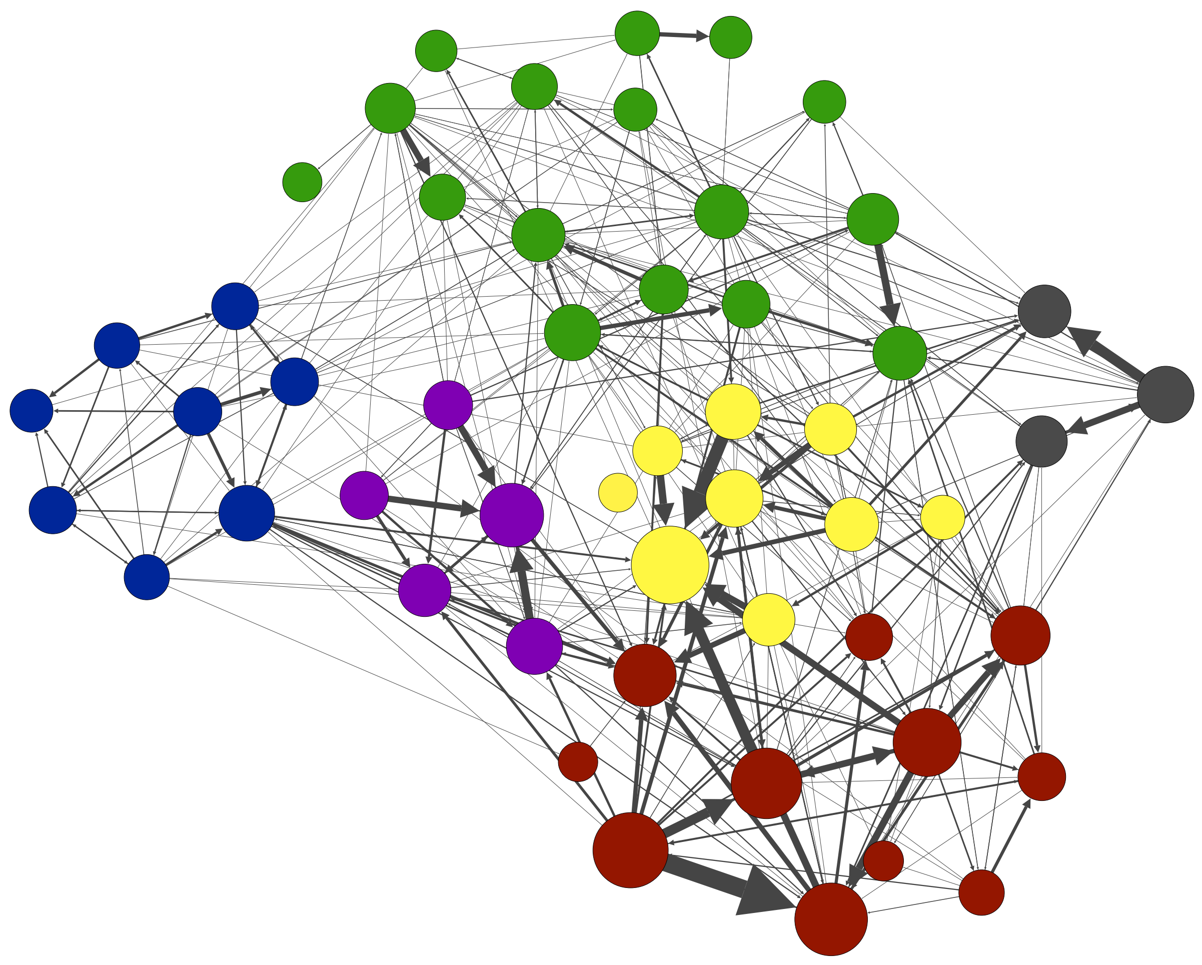


Figure 7.17: *Sub-networks* inside the *Communication network* of $cluster_5$. The thinnest edge indicates 350 calls, and and size of the *vertex* corresponds to *betweenness centrality* of each user. *Vertices* with the same color belong to the same *sub-networks*.

User Behavior Pattern 3: Medium Level Receiver

Users of one community ($cluster_{14}$) follow this behavioral pattern. These users mainly receive a large number of calls (at least ten times higher number of calls) than they generate. Though these users usually receive more call than they dial, the in-coming call volume is not too high. On average they receive less than 200 calls per year. That is why, we define these users as *medium level receiver*.

Though, here a few users generate calls, most of their calls are directed to other clusters, "internal inactive" and "external" user groups. That is why, users of $cluster_{14}$ are very lightly connected with each other. This indicates that users of this community are grouped together based on their similarity of common friends from whom they receive a large number of calls. Fig. 7.18 shows the internal communication between users of $cluster_{14}$ (161 users). It shows that, only 8% of the users communicate with each other for maximum 8 times during one year.

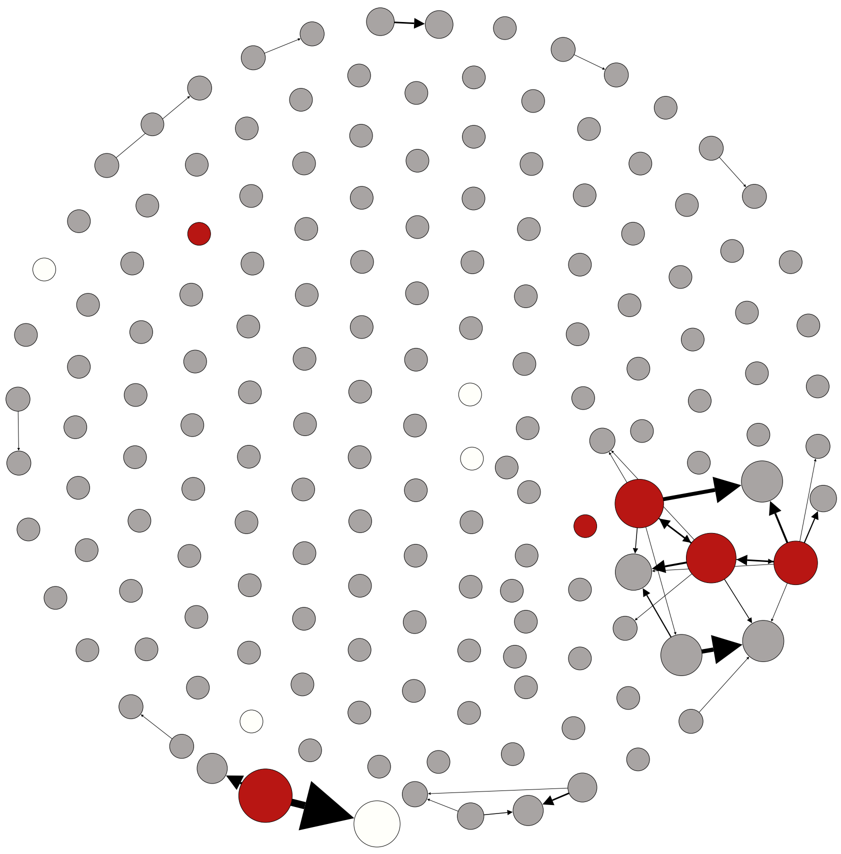


Figure 7.18: Internal communication of users inside $cluster_{14}$ (Thinnest edge indicates 1 call). Red filled "circles" indicate callers, grey filled "circles" indicate receivers, and, white "circles" indicate both caller and receiver. Size of the *vertex* corresponds to the frequency of dialed & received calls by users.

Most of the users of $cluster_{14}$ remain active at least for 60% time of the observation window. To understand the structure of social relationships of users of this community (*cluster*$_{14}$), we consider the *communication network* of *cluster*$_{14}$, CN_{14} which consists of $10,176$ users (161 members of *cluster*$_{14}$ and $10,015$ neighbors). We find out that this communication networks is very sparse. About 80% of the neighbors are "external" users and a large number of them appears very few times during the observation window. Only the 109 of 10,176 users (of the communication network of *cluster*$_{14}$) communicated more than 50 times in the observation window. In fact, these 109 users (consists of a few members of *cluster*$_{14}$, other "internal" and "external" users) have created a strongly connected network inside CN_{14}. Fig. 7.19 shows that these most active 109 users of CN_{14} build sort of a *star* network where one user of *cluster*$_{14}$ has strong connection to most other users. We suspect that this central node probably a "shared number" or a fax machine server. Fig. 7.20 shows that, users of CN_{14} can be divided into 2 *sub-networks* based on the communication pattern.

User Behavior Pattern 4: High Volume Receiver

Users of four different communities/clusters (cluster 3, 10, 11, and, 12) follow this behavioral pattern. The main characteristic of the communication pattern of these users is that they receive a huge amount of calls than they generate during the observation window. About 90% users receive at least hundred times higher number of calls than they generate. On average they receive more than 400 calls while generate less than 20 calls per year.

Though, users of these clusters receive a huge number of calls, most of their calls are generated by other groups of "internal" and "external" users. That is why, users inside these clusters are very lightly connected. Fig. 7.21 shows the internal communication between users of *cluster*$_{12}$. It is noticed that users of *cluster*$_{12}$ are not connected at all, this indicates that users of this cluster are grouped together based on their similarity of

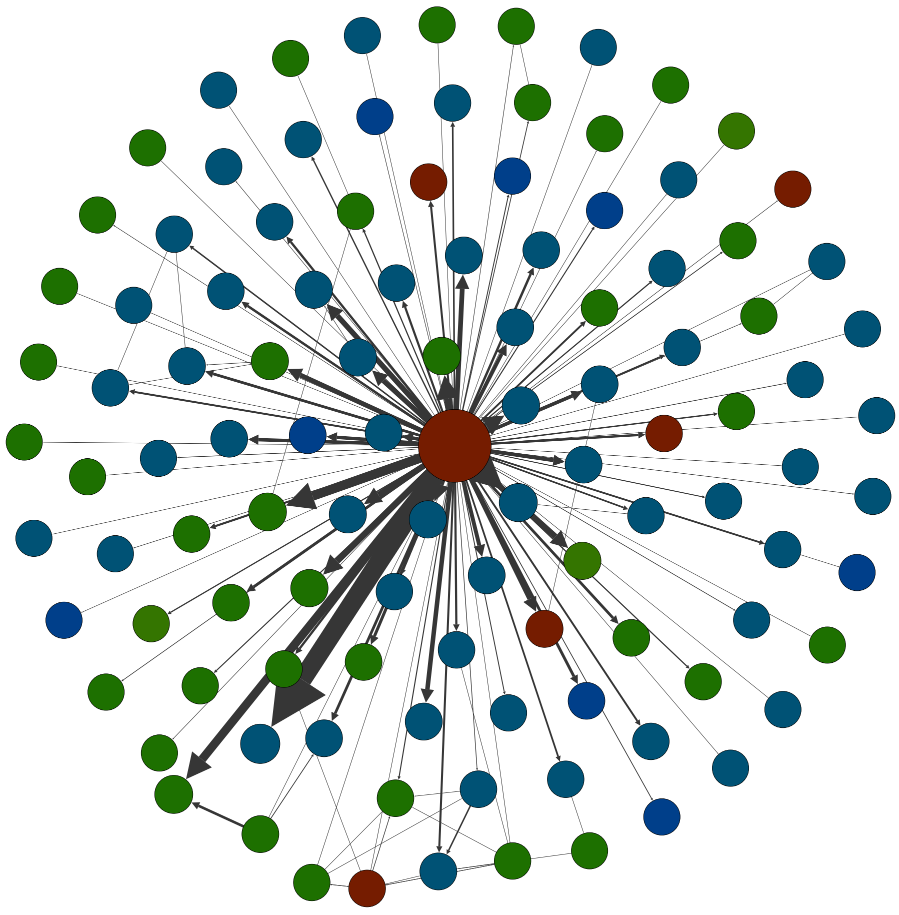


Figure 7.19: *Communication network* of $cluster_{14}$, CN_{14}. The thinnest edge indicates 50 calls and size of the *vertex* corresponds to the frequency of dialed & received call by each user. Red "circles" indicate members of $cluster_{14}$, green "circles" indicate *internal* users, and, blue "circles" indicate *external* users

common friends from whom they received a huge amount of calls during the observation window.

Most of the users of these clusters remain active at least for 60% time of the observation window. The average duration of the dialed and received calls by users of these clusters is 4 minutes. We also consider how users of these three clusters communicate with other users (both "internal" and "external"). To this end, we consider the *communication network* of each

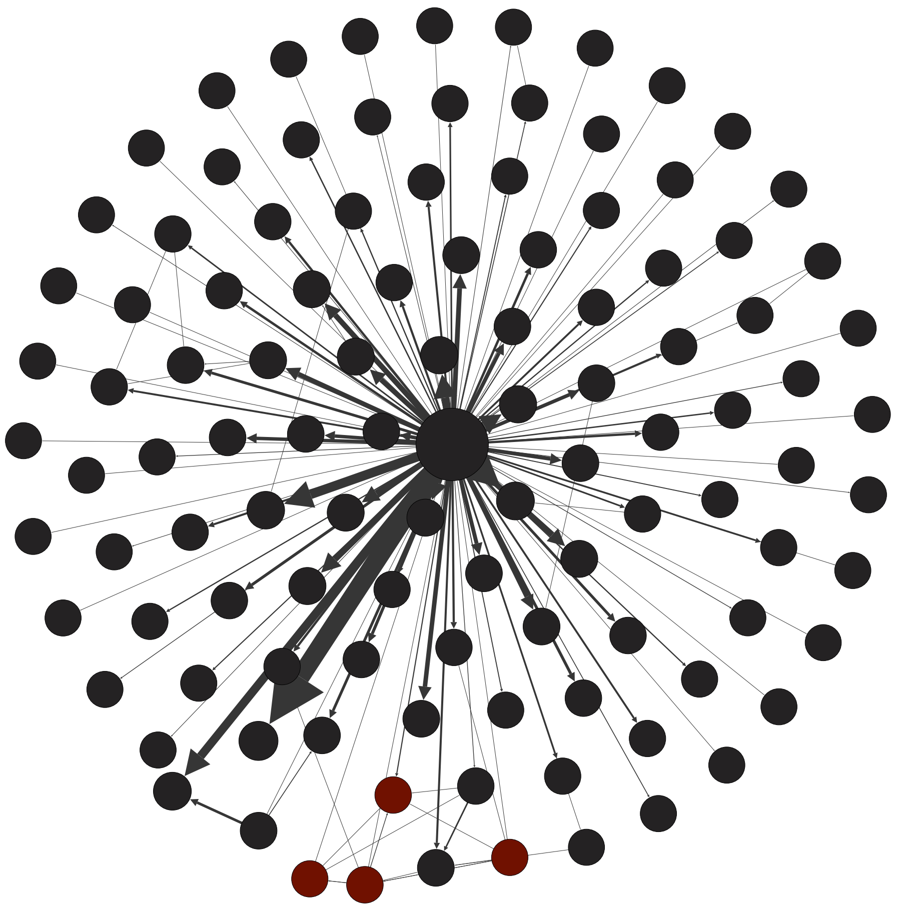


Figure 7.20: *Sub-networks* inside the *Communication network* of $cluster_{14}$ (Thinnest edge indicates 50 calls, colors and size of the vertex corresponds to the internal sub-communities, and *betweenness centrality* of each user). *Vertices* with the same color belong to the same *sub-networks*.

of the four clusters. For instance, the *communication network* of $cluster_{12}$, CN_{12} consists of 6363 users (131 members of this cluster and 6232 neighbors who are communicated with members of $cluster_{12}$ during the observation window). From CN_{12} it is noticed that around 63% of the communicated users are *external*. But, most of them appear very few times during the observation window. If we exclude the users that generate or receive less than 80 calls per year from CN_{12}, then we find a well-connected network

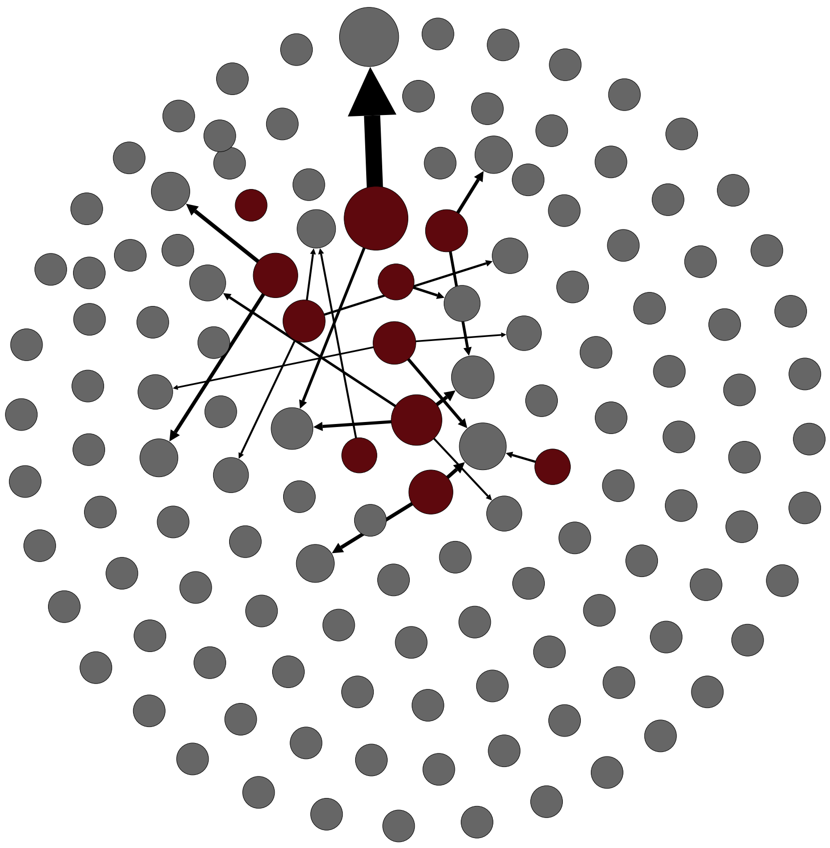


Figure 7.21: Internal communication of users inside *cluster*$_{12}$ (Thinnest edge indicates 5 calls). Red filled "circles" indicate callers and grey filled "circls" indicate receivers. Size of the *vertex* corresponds to the frequency of dialed & received called by user.

which consists of only 114 of 6363 users. Fig. 7.22 shows the social interaction of these 114 users in CN_{12}. These active users form 5 *sub-networks* inside the CN_{12} based on their social relationship with each other (shown in Fig. 7.23).

User Behavior Pattern 5: Both Caller & Receiver

Users of three communities (*cluster*$_2$, *cluster*$_3$ and *cluster*$_{15}$) follow this behavioral pattern. These users do not have any significant characteristics, they usually both dial and receive a large number of calls, on average

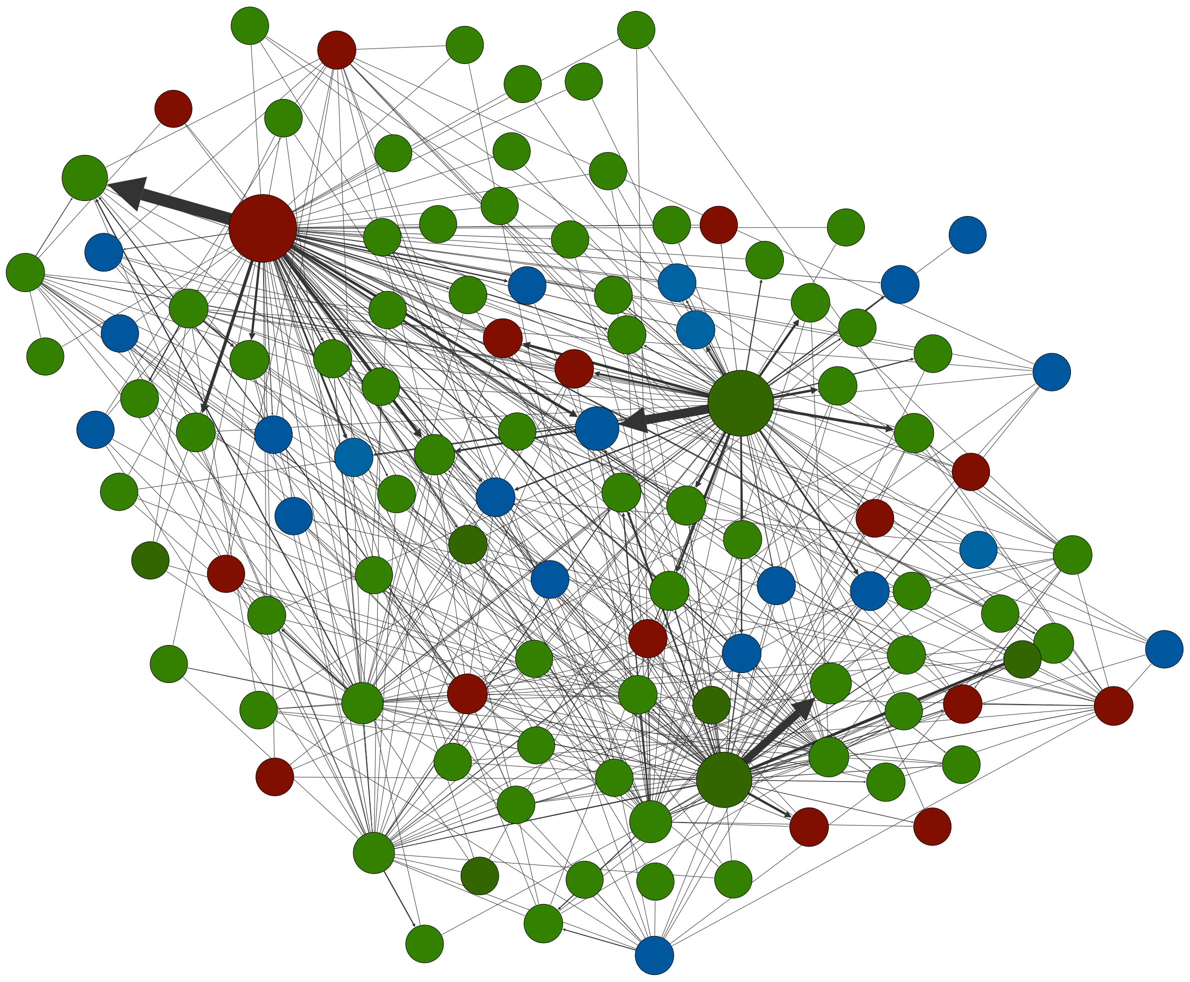


Figure 7.22: *Communication network* of $cluster_{12}$, CN_{12}. The thinnest edge indicates 80 calls, size of the vertex corresponds to the frequency of dialed & received call by each user. Red "circles" indicate members of $cluster_1$, green "circles" indicate *internal* users, and, blue "circles" indicate *external* users.

they receive around 250 calls and dial around 200 calls per year. 60% of their communicated users are from other "internal" and "external" groups of users, while, the remaining 40% of their calls are directed to the users of the same cluster. That is why, internal communications are observed inside these three communities. This indicates that users of these three communities are grouped together based on both their similarity of common friends

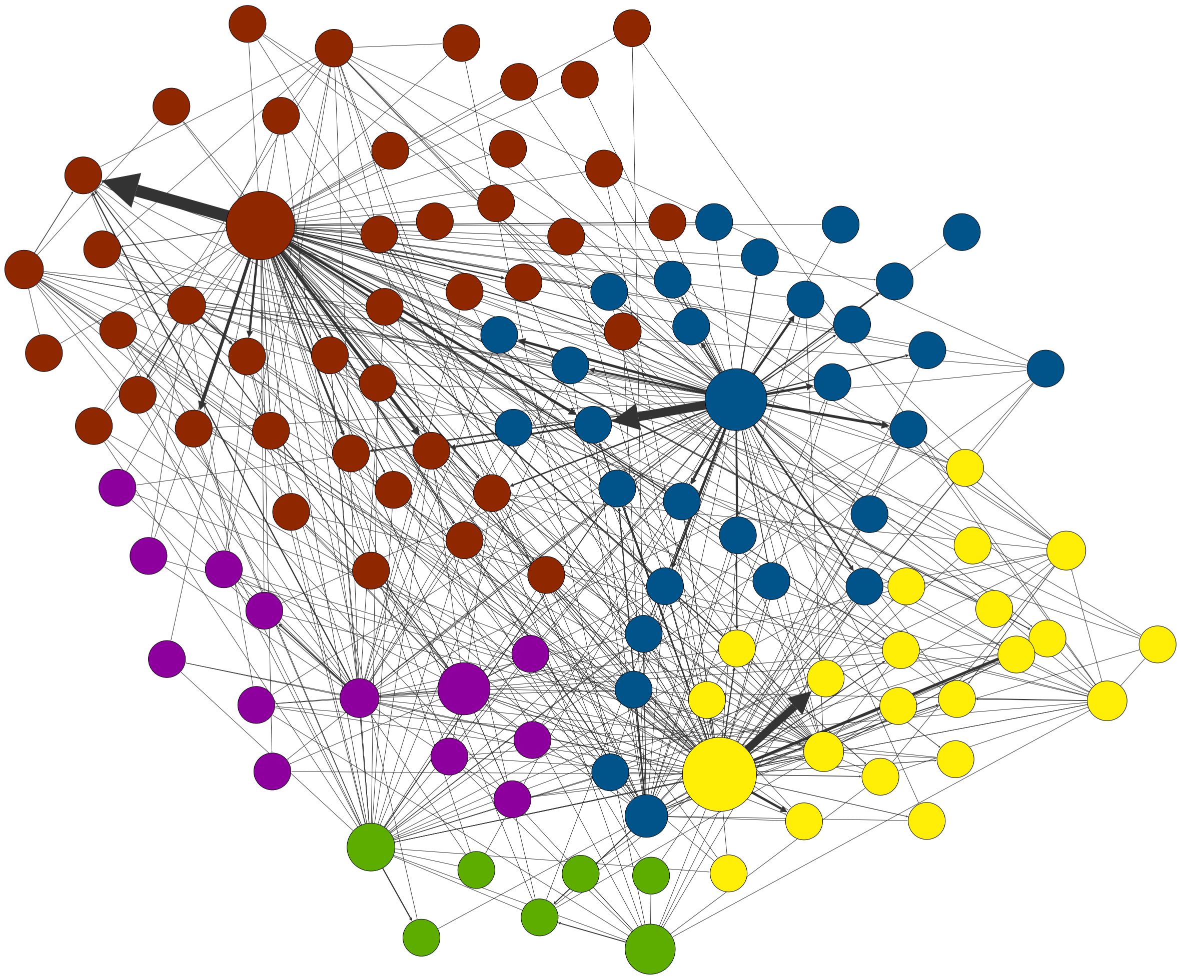


Figure 7.23: *Sub-networks* inside the *Communication network* of $cluster_{12}$ (Thinnest edge indicates 80 calls, colors and size of the vertex corresponds to the internal sub-communities, and *betweenness centrality* of each user). *Vertices* with the same color belong to the same *sub-networks*.

and the frequency of their internal communication. For instance, Fig. 7.24 shows the internal communication between users of $cluster_3$. It shows the 37% of the users of this community that has communicated with each other more than 20 times during a year.

Most of the users of $cluster_3$ remain active at least for 80% time of the observation window. The average duration of the dialed and received calls

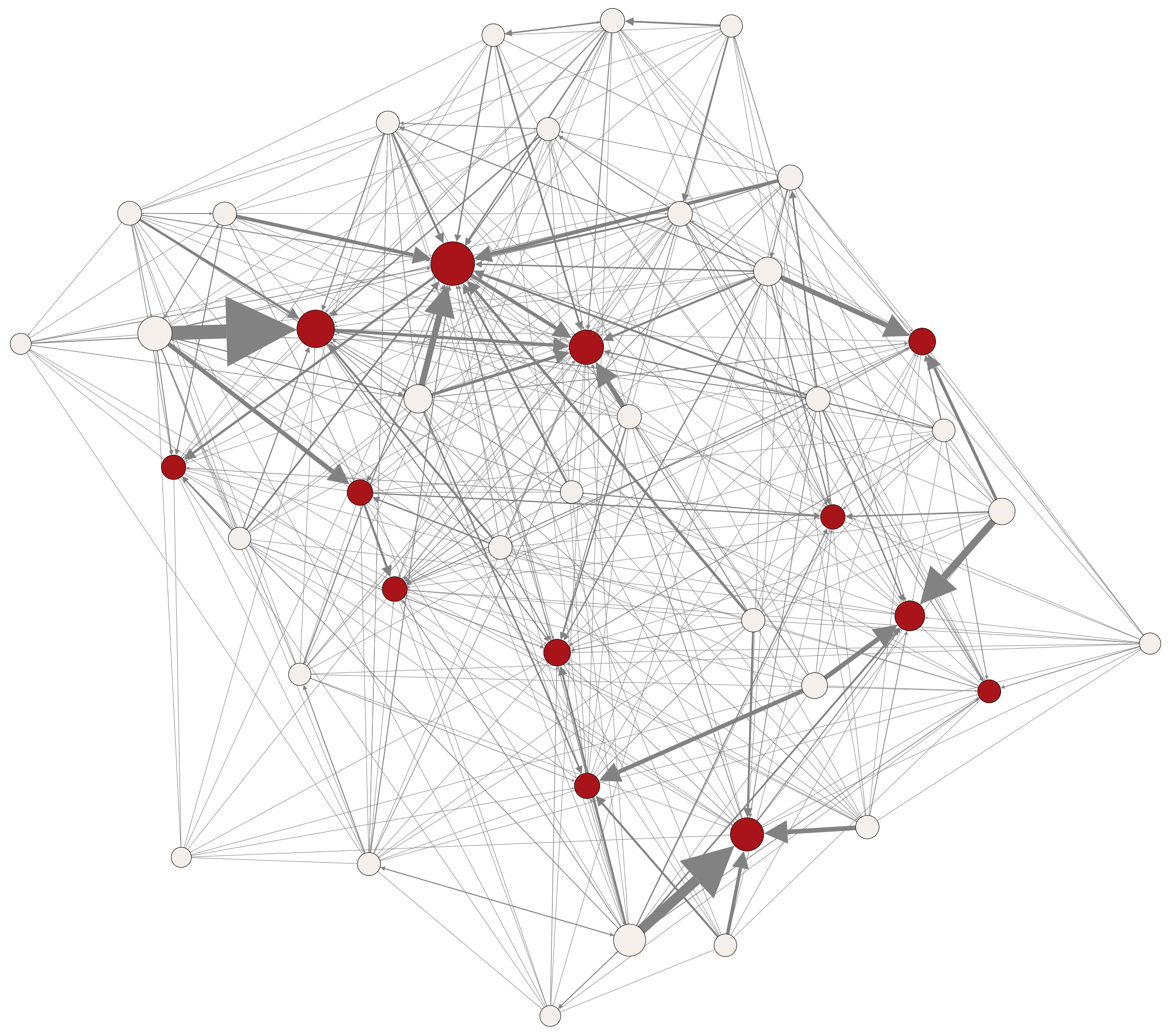


Figure 7.24: Internal communication of users inside *cluster*$_3$ (Thinnest edge indicates 20 calls). Red filled "circle" indicates callers and white "circle" indicates both caller and receiver. Size of the *vertex* corresponds to the frequency of dialed & received called by user.

by users of these clusters is 4 minutes. Most of the calls generated and received by users of these clusters are during peak hour of working days.

The *communication network* of *cluster*$_3$, CN_3 consists of 4992 users (119 members of *cluster*$_3$ and 4873 neighbors who are communicated with members of *cluster*$_3$ during the one year observation window). At first glance, we find CN_3 to be very sparse due to the fact that around 60% of the communicated users are "external". That is why, around 50% members of *cluster*$_3$ appears to be very *social* (e.g., these users tend to dial and receive calls from a large variety of people). Later it is noticed that most of these "external" users appear very few times during the observation window. If we exclude the users that generate or receive less than 100 calls per year from CN_3, then we find a well-connected network of 48 users and all of them are members of *cluster*$_3$. Fig. 7.25 shows the communication pattern of these 48 of 4992 users in CN_3.

Based on their call pattern, these 48 users are further divided into 5 strong *sub-networks*. This indicates that though each user of *cluster*$_3$ communicated with a large number of users (both "internal" and "external"), in the end, the most active and well-connected users maintain a regular communication with a small number of people within the same cluster. Fig. 7.26 shows the *sub-networks* of 48 users inside CN_3 and also the betweenness centrality of each users.

7.3 Detection of *Malicious* Users based on Social Interaction Pattern

Analysis of VoIP traffic and social interaction pattern of users can assist the detection of *malicious* users. We have implemented a prototype of *malicious* users detection system where the detection engine is a SVM classifier. The information retrieved from the social behavior analysis of VoIP users is used to train the classifier. The logical architecture of the *malicious* message detection engine is shown in Fig. 3.2 in the "VoIP Call Filtering Module". This section demonstrates the performance of the that filtering module. This module controls the well-formed SIP messages (that are successfully passed the *syntax* and *semantic* error checking) to detect

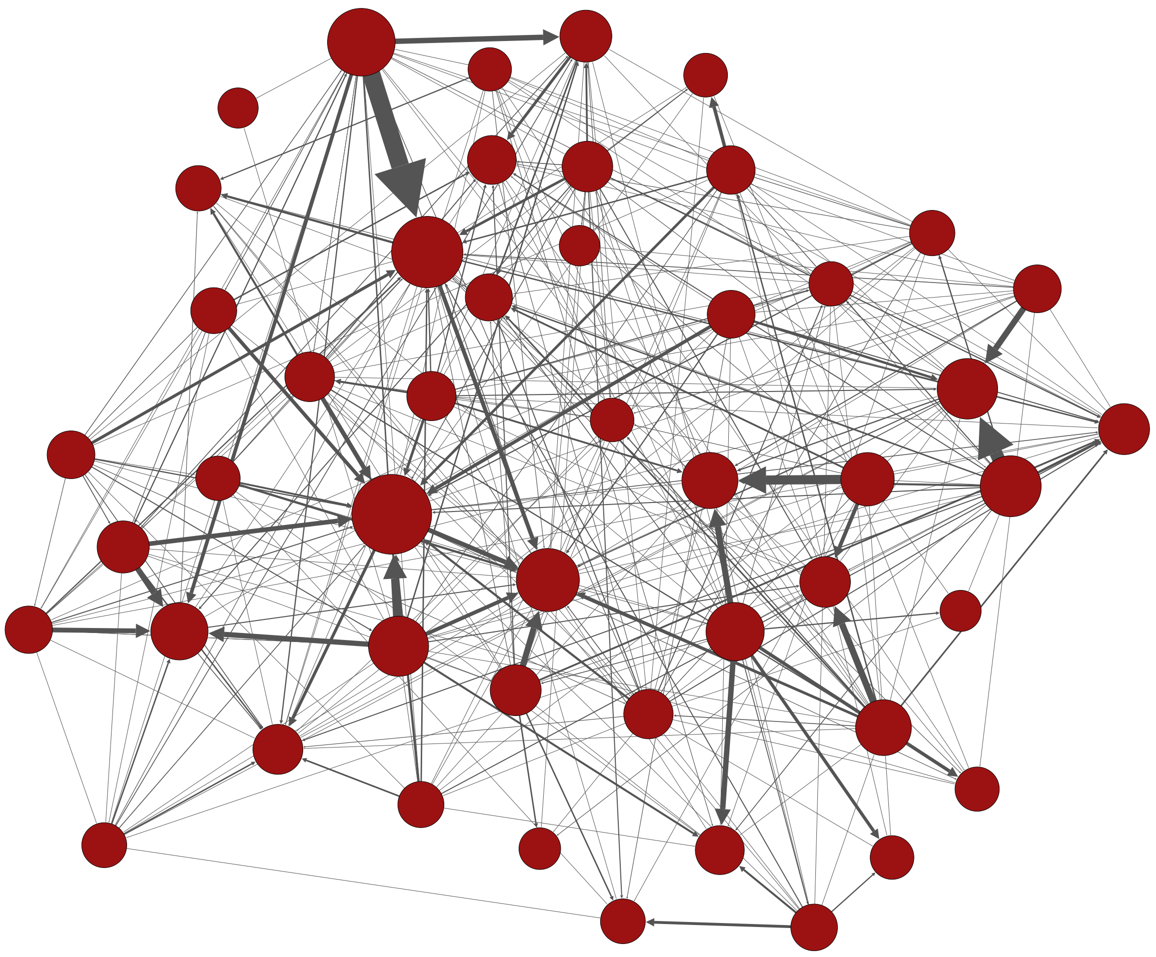


Figure 7.25: *Communication network* of *cluster*$_3$ (Thinnest edge indicates 100 calls, size of the vertex corresponds to the frequency of dialed & received call by each user). Red "circles" indicate members of *cluster*$_3$.

the presence of *malicious* content and users.

7.3.1 Training of SVM Classifier

Application of social network analysis techniques in the stream of SIP messages (captured from our institution) reveals normal behavioral pattern of the "internal" users (both "active" and "inactive") in the system (described in Section 7.2). Specially, five behavioral patterns of the "internal active"

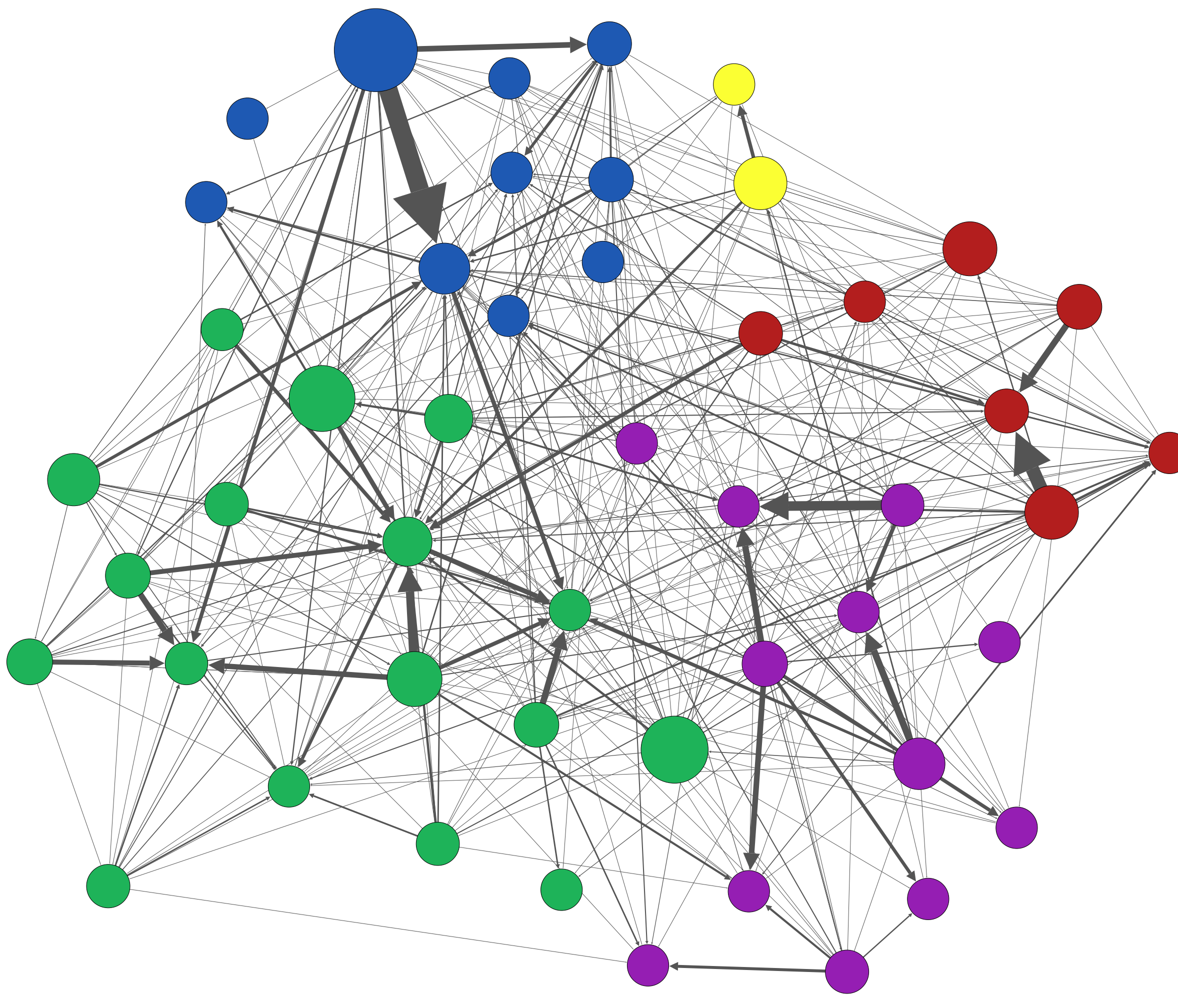


Figure 7.26: *Sub-networks* inside the *Communication network* of *cluster*$_3$ (Thinnest edge indicates 100 calls, colors and size of the vertex corresponds to the internal sub-communities, and *betweenness centrality* of each user). *Vertices* with the same color belong to the same *sub-networks*.

users are noticed and we consider all of them to be "legitimate"[4].

The SVM classifier is trained with the "legitimate" behavior patterns of "active" and "inactive" internal users. The classifier, thus, learned about the behavior of "internal" users, examines the unlabeled call vectors in the

[4]VoIP network of our institution being a closed network, the probability of transmitting "harmful" messages is low.

"test" set. During this process, the classifier observes the behavior of the "internal" users and any suspicious deviation of their behavior is identified.

7.3.2 Data Set

The performance of the "VoIP Call Filtering Module" of the SIP message filtering system proposed in section 3.2 is evaluated through various experiments. For these experiments we have used a sub-set of the collected SIP traces from our institution. The sub-set of the collected sample consists of six months (October 2013- March 2014) SIP trace with around 96 millions SIP messages. The SIP messages are further processed and $422,687$ calls are retrieved from the 96 million SIP messages among which $230,816$ were successfully established. Each call is represented with the information about the arrival time, duration, caller and receiver name of the call which is known as the CDR (call detail record). Around 48 thousand SIP URIs are found that are related to these calls, among these $8,498$ are *internal* users while the remaining are *external* users.

Dataset collected from our institution do not contain any messages that were maliciously sent with the intention of harming the network due to the fact that this is a closed network. To counteract the lack of "malicious" calls in our collected sample, we randomly inject synthetic "malicious" calls into the stream of real VoIP calls. The synthetic attack traces are generated using our developed synthetic traffic generator "VoIPTG" (described in Section 5.3).

Injection of Synthetic Malicious Traces

Though there are various kinds of "attacks" listed in the literature, in this experiment, we have focused on detection of *social threats* (e.g., spam calls or SPIT) and toll fraud.

We focus on these two categories of attacks as detection of these silent attacks is difficult. For instance, the characteristics of the spam callers are not always clearly separable from that of the "legitimate" users. Usually, telemarketers and advertisers tend to call a large number of people to deliver their messages [8,10,32,41,106,108,165,169] often resulting in short duration calls as the other party quickly hangs up after realizing that the call is spam. Thus, the spam callers have high out-going degree with very low or almost zero incoming degree as usually they receives very limited calls or no call at all. On the other hand, traffic analysis of real trace collected from our institution shows that a large number of "legitimate" users only generate calls while receives very few or no call at all during the whole observation period. In such context, it requires a delicate handling to distinguish the group of "legitimate" callers from the group of *spammers*.

Inorder to imitate *social attack* scenario, we inject a huge amount of "spam" calls randomly selected 110 "internal" (70 "active" and 40 "inactive") users among 8,498 users found in the dataset containing real VoIP calls. Following the general assumption about the characteristics of the "spam" calls, these synthetic calls are generated during peak hour, with very low interarrival time difference and of short duration. Callers of these calls are a set of synthetically generated "external" SIP uris that refers to the set of spam callers.

Toll fraud attack refers to the scenario where due to the deployment of inadequate security, "malicious" users can make unauthorized usage of paid communication services (such as international calling) where they may call few subscribers ("legitimate" users) of the network and prepended a "malicious" number (mostly in some other country) to which the system sets up calls. In result, the organization has to pay a huge phone bill that is way higher than its typical monthly bill. In order to imitate such scenario, we randomly select 60 "internal" users (30 "active" and 30 "inactive"). A

Table 7.4: Description of synthetic attacks scenarios

Scenario	Description	Time	Total Call	"Internal" Users	"Malicious" Users
"Spam" calls	2 week (weekdays)	8 am-5 pm	40,000	30	30
"Spam" calls	1 week (weekdays)	8 am-12 pm	30,000	40	30
"Spam" calls	1 week (weekdays)	11 am-6 pm	30,000	40	30
"Toll Fraud"	4 week (weekdays & weekends)	8 am-11 pm	80,000	40	30
"Toll Fraud"	2 week (weekdays)	8 am-5 pm	20,000	20	30

large number of synthetic calls are injected into the real trace indicating that these calls are generated from these 60 users to a set of external SIP URIs (indicating "malicious" numbers). Details about the synthetic attack scenarios are reported in Table 7.4.

Detail about the new "test" set (combination of real and synthetic calls) is found in Table 7.5.

7.3.3 Performance Evaluation

Detail about achieved accuracy by the SVM classifier on the "test" set is found in Table 7.5. The SVM classifier checks the behavior of all the $8,498$ internal users (observed in the test set). Total 1,020 "internal" users are found those were not present during the training period. The classifier declares them as "good" user as they follow the behavioral pattern of "legitimate" users during the observation window. On the other hand, behavior of 426 users are identified as "suspicious" due to the fact that these users suddenly become "active" during "test" period, while they were "inactive" during training. These users are "legitimate".

All our experiments are done in a machine of Intel Core i7 CPU, 2.0 GHz Quad-core and 8 GB RAM memory. We have also focused on the time required for the SVM classifier to detect an attack scenario. This classifier observes every call and checks the behavior of the "internal" users associ-

Table 7.5: SIP call classification result using SVM

Description	Value
Total Calls	430,816
Number of real "good" calls	230,816
Number of synthetic "spam" calls	100,000
Number of synthetic "toll fraud" calls	100,000
Total Number of Users observed in test set	48,699
Number of "internal active" users found in test set	1,985
Number of "internal inactive" users found in test set	6,513
Number of "external active" users found in test set	412
Number of "external inactive" users found in test set	39,789
New "internal active" users not found during training	33
New "internal inactive" users not found during training	987
Number of "internal" users who receive synthetic "spam" calls	110
Number of "internal" users who generate synthetic "toll fraud" calls	60
SVM Classifier Result	
True positive("malicious" users identified as "malicious")	161
False positive("good" users identified as "malicious")	426
True negative("good" users identified as "good")	7,902
False negative("malicious" users identified as "good")	9
Accuracy	94.88%

ated with that call. As soon as it notices any significant deviation of the behavior of any of the "internal" users from his/her "normal" behavior, the classifier suspects that user as "malicious". In the synthetic attack scenarios (described in Table 7.4), the attack starts slowly from zero and increases until it reaches the maximum rate, then, the maximum rate is maintained constant until the attack duration stops. We noticed that the SVM classifier is able to detect the attacks as soon as the traffic reaches the maximum rate. For example, during synthetic "social" attack scenario, huge amount of *spam* calls (at least 100 call/day) are directed to an "internal" users for five consecutive days. This changes the victim user's behavior significantly in the context of receiving calls as he/she normally receive at most 10/20 calls per day. The classifier observes each call directed to this user and

Table 7.6: Experiments performed on traces generated by "Spitter" tool

Description	Value
Total Calls	5,000
Number of "good" calls	3,128
Number of "spam" calls	1,872
Total Number of "good" users	100
Total Number of "malicious" users	20
SVM Classifier Result	
True positive("malicious" users identified as "malicious")	18
False positive("good" users identified as "malicious")	3
True negative("good" users identified as "good")	97
False negative("malicious" users identified as "good")	2
Accuracy	95.83%

it notices a dramatic increase in the frequency of received calls by this user. At the end of the day, considering also the patterns (low duration and interarrival time) of those incoming calls towards the *victim* user, the classifier suspects that this user is receiving a lot of "spam" calls.

7.3.4 Experiment with "Spitter" tool

For the purpose of the evaluation of the performance and efficiency of our proposed filtering system for SIP based VoIP networks with other attack simulators, we have used "Spitter"[5] which a publicly available tool implemented by the author of [48] to perform VoIP spam testing. This tool uses the Asterisk IP PBX as a platform from which it launches SPIT calls (i.e. VoIP SPAM). The input file of "Spitter" contains one or more Asterisk ASCII call "records" [191]. In order to avoid the risk of directly using this tool to transmit spam calls through the SIP servers of our institution, we followed an alternative way. We use the tool in *test mode* which allows the generation of spam call records in the absence of an Asterisk installation. The generated synthetic spam calls are injected the a set of "good" VoIP

[5] "Spitter" - `http://www.hackingvoip.com/tools/spitter.tar.gz`

call records extracted from a sub-set of collected real SIP traces. Details about dataset generated by the tool “Spitter” and performance evaluation of our classifier with this dataset is found in Table 7.6.

Chapter 8

Conclusion

Voice over IP (VoIP) has become a major paradigm for providing flexible telecommunication services while reducing operational and maintenance costs. Being a fast-growing Internet application, VoIP shares the network resources with the regular Internet traffic, and is susceptible to the existing security holes of the Internet. Moreover, given that voice communication is time sensitive and uses a suite of interacting protocols (e.g., SIP, RTP), VoIP exposes to new forms of vulnerabilities specific to these protocols and real time communication constraints. Thus, security of VoIP communications has become increasingly important.

This thesis has contributed to the field of security of SIP based VoIP in several ways, looking both into general issues like defining a formal representation of VoIP related security issues and more specific topics like to design and implementation of traffic analysis and filtering systems.

The first contribution is a filtering system for the conformance and security control of the SIP based VoIP services. This proposed filtering system can be considered as a second line of defense of SIP based VoIP networks to detect the violation of existing security policies. It analyzes the stream of incoming and outgoing SIP messages of the network to discard the messages with *syntax* and *semantic* errors. Considering the difference in structure, content and timing of the “erroneous” SIP messages, the proposed filtering system consists of a multistage classifiers. The first stage controls the validity of the message syntax through a deterministic and efficient process. While first stage filtering is straightforward, as the classification is crisp (either a messages belongs to the language or it does not), the second stage requires a more delicate handling, as it is not a sharp decision whether a message is semantically meaningful or not. The approach we followed for this step is based on using past experience on previously classified messages, i.e. a “learn-by-example” approach, which led to classifiers based on Support-Vector-Machines (SVM) to perform the required analysis of each incoming SIP message. The advantages of the SVM is that very few parameters are required for tuning the learning machine and a small sample set can build the model to classify huge unlabeled dataset. It is observed that careful selection and configuration of the kernels of the SVM is utmost important for the efficient implementation of the classifier, as well as on its training to achieve a high accuracy in the classification process. The efficiency of the filtering system tested both with real and artificial traces is very promising. In an effort to reduce the computational complexity of the SVM classifier, additional analyses showed that virtually the same accuracy could be achieved even when reducing the size of the set of feature vectors through principal component analysis.

The second contribution has considered the use of Social Network Analysis (SNA) techniques to explore the social behavioral patters of VoIP users. “Social networks” of VoIP users are build from their call records for a large observation window (e.g., one and a half years). We have shown the influence of the analysis of social groups of VoIP users in application of security policies and detection of *malicious* users. To this end, an extra stage is added to our proposed multistage filtering architecture. This stage consists of a Support Vector Machine (SVM) classifier which “learns” the social behavioral profiles of the *legitimate* users through an initial training phase and, the SVM, thus configured correctly is able to detect *malicious* in the VoIP traffic.

Finally, we have introduced “VoIPTG”, a generic traffic simulator, which is capable to generate traffic following various possible models of SIP based VoIP system with special attention to characterize the sophisticated user behavior in the system. This simulator is implemented to address the lack of a publicly available dataset containing VoIP traces for testing security control systems. This stochastic generator focuses on modeling the realistic behavior profiles for users and attackers, and, thus provides flexibility and efficiency in generation of large amount of synthetic traffic of SIP based VoIP system. To generate the message level VoIP traffic, we have implemented “SIP-Msg-Gen”, a state-full and context aware SIP fuzzer, which is capable of generating both “good” and “bad” SIP messages.

Bibliography

[1] Humberto J Abdelnur, Olivier Festor, et al. KiF: a stateful SIP fuzzer. In *Proceedings of the 1st international conference on Principles, systems and applications of IP telecommunications*, pages 47–56. ACM, 2007.

[2] Fateme Abdoli and Mohsen Kahani. Ontology-based distributed intrusion detection system. In *Computer Conference, 2009. CSICC 2009. 14th International CSI*, pages 65–70. IEEE, 2009.

[3] Muhammad Ali Akbar and Muddassar Farooq. Securing SIP-based VoIP infrastructure against flooding attacks and Spam over IP Telephony. *Knowledge and Information Systems*, pages 1–20, 2012.

[4] Ronnie Alves, Pedro Ferreira, Orlando Belo, Joao Lopes, Joel Ribeiro, Luís Cortesão, and Filipe Martins. Discovering telecom fraud situations through mining anomalous behavior patterns. In *Proceedings of the DMBA Workshop, on the 12th ACM SIGKDD*. Citeseer, 2006.

[5] Jari Arkko, Gonzalo Camarillo, Aki Niemi, Tao Haukka, and Vesa Torvinen. Security mechanism agreement for the Session Initiation Protocol (SIP). (RFC 3329), 2003.

[6] Stefan Axelsson. Intrusion Detection Systems: A Survey and Taxonomy. Technical report, 2000.

[7] Muhammad Ajmal Azad and Ricardo Morla. Multistage SPIT detection in transit VoIP. In *Software, Telecommunications and Computer Networks (SoftCOM), 2011 19th International Conference on*, pages 1–9. IEEE, 2011.

[8] Muhammad Ajmal Azad and Ricardo Morla. Caller-REP: Detecting unwanted calls with caller social strength. *Computers & Security*, 39:219–236, 2013.

[9] Yan Bai, Xiao Su, and Bharat Bhargava. Adaptive voice spam control with user behavior analysis. In *High Performance Computing and Communications, 2009. HPCC'09. 11th IEEE International Conference on*, pages 354–361. IEEE, 2009.

[10] Vijay Balasubramaniyan, Mustaque Ahamad, and Haesun Park. CallRank: Combating SPIT Using Call Duration, Social Networks and Global Reputation. In *CEAS*, 2007.

[11] Greg Banks, Marco Cova, Viktoria Felmetsger, Kevin Almeroth, Richard Kemmerer, and Giovanni Vigna. SNOOZE: toward a Stateful NetwOrk prOtocol fuzZEr. In *Information Security*, pages 343–358. Springer, 2006.

[12] Joachim Baumeister and Dietmar Seipel. Anomalies in ontologies with rules. *Web Semantics: Science, Services and Agents on the World Wide Web*, 8(1):55–68, 2010.

[13] Richard A Becker, Chris Volinsky, and Allan R Wilks. Fraud detection in telecommunications: History and lessons learned. *Technometrics*, 52(1), 2010.

[14] Richard Benjamins, Dieter Fensel, and Asunción Gómez-Pérez. Knowledge management through ontologies. CEUR Workshop Proceedings (CEUR-WS. org), 1998.

[15] Giuseppe Bianchi, Nico d'Heureuse, and Saverio Niccolini. On-demand time-decaying bloom filters for telemarketer detection. *ACM SIGCOMM Computer Communication Review*, 41(5):5–12, 2011.

[16] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, 2008.

[17] Hossein Kaffash Bokharaei, Alireza Sahraei, Yashar Ganjali, Ram Keralapura, and Antonio Nucci. You can SPIT, but you can't hide: Spammer identification in telephony networks. In *INFOCOM, 2011 Proceedings IEEE*, pages 41–45. IEEE, 2011.

[18] Richard J Bolton, David J Hand, et al. Unsupervised profiling methods for fraud detection. *Credit Scoring and Credit Control VII*, pages 235–255, 2001.

[19] Stephen P Borgatti and Martin G Everett. A graph-theoretic perspective on centrality. *Social networks*, 28(4):466–484, 2006.

[20] Yacine Bouzida and Christophe Mangin. A framework for detecting anomalies in VoIP networks. In *Availability, Reliability and Security, 2008. ARES 08. Third International Conference on*, pages 204–211. IEEE, 2008.

[21] Ulrik Brandes. A faster algorithm for betweenness centrality*. *Journal of Mathematical Sociology*, 25(2):163–177, 2001.

[22] Peter Burge and John Shawe-Taylor. An unsupervised neural network approach to profiling the behavior of mobile phone users for use in fraud detection. *Journal of Parallel and Distributed Computing*, 61(7):915–925, 2001.

[23] Christopher J.C. Burges. A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.

[24] David Butcher, Xiangyang Li, and Jinhua Guo. Security challenge and defense in VoIP infrastructures. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 37(6):1152–1162, 2007.

[25] Rich Caruana and Alexandru Niculescu-Mizil. An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd international conference on Machine learning (ICML '06)*, pages 161–168, New York, USA, 2006.

[26] Noppawat Chaisamran, Takeshi Okuda, Gregory Blanc, and Suguru Yamaguchi. Trust-based VoIP spam detection based on call duration and human relationships. In *Applications and the Internet (SAINT), 2011 IEEE/IPSJ 11th International Symposium on*, pages 451–456. IEEE, 2011.

[27] ChihChung Chang and ChihJen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:1–27, 2011.

[28] Olivier Chapelle and Vladimir Vapnik. Model Selection for Support Vector Machines. In *Advances in neural information processing systems*, volume 12, pages 230–236, 1999.

[29] Eric Y Chen. Detecting dos attacks on SIP systems. In *VoIP Management and Security, 2006. 1st IEEE Workshop on*, pages 53–58. IEEE, 2006.

[30] Whai-En Chen, Hui-Nien Hung, and Yi-Bing Lin. Modeling VoIP call holding times for telecommunications. *Network, IEEE*, 21(6):22–28, 2007.

[31] Zhen Chen, Shize Guo, Kangfeng Zheng, and Haitao Li. Research on man-in-the-middle denial of service attack in SIP VoIP. In *Networks Security, Wireless Communications and Trusted Computing, 2009. NSWCTC'09. International Conference on*, volume 2, pages 263–266. IEEE, 2009.

[32] S. Chiappetta, C. Mazzariello, R. Presta, and S.P. Romano. An anomaly-based approach to the analysis of the social behavior of VoIP users. *Computer Networks*, 2013.

[33] Aaron Clauset, Mark EJ Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical review E*, 70(6):066111, 2004.

[34] Kenneth C Cox, Stephen G Eick, Graham J Wills, and Ronald J Brachman. Brief application description; visual data mining: Recognizing telephone calling fraud. *Data Mining and Knowledge Discovery*, 1(2):225–231, 1997.

[35] David Crocker and Paul Overell. Augmented BNF for Syntax Specifications: ABNF. (RFC 4234), 2005.

[36] John N Daigle and Joseph D Langford. Models for analysis of packet voice communications systems. *Selected Areas in Communications, IEEE Journal on*, 4(6):847–855, 1986.

[37] Trang Dinh Dang, Balázs Sonkoly, and Sándor Molnár. Fractal analysis and modeling of VoIP traffic. In *Telecommunications Network Strategy and Planning Symposium. NETWORKS 2004, 11th International*, pages 123–130. IEEE, 2004.

[38] Ram Dantu and Prakash Kolan. Detecting spam in VoIP networks. In *Proceedings of the steps to reducing unwanted traffic on the internet on steps to reducing unwanted traffic on the internet workshop*, pages 5–5. USENIX Association, 2005.

[39] P. A. Devijver and J. Kittler. *Pattern Recognition: A Statistical Approach.* Prentice Hall, London, GB, 1982.

[40] Nico d'Heureuse, Jan Seedorf, and Saverio Niccolini. A policy framework for personalized and role-based SPIT prevention. In *Proceedings of the 3rd International Conference on Principles, Systems and Applications of IP Telecommunications*, page 12. ACM, 2009.

[41] Nico d'Heureuse, Sandra Tartarelli, and Saverio Niccolini. Analyzing telemarketer behavior in massive telecom data records. In *Trustworthy Internet*, pages 261–271. Springer, 2011.

[42] Tim Dierks. The transport layer security (TLS) protocol version 1.2. (RFC 5246), 2008.

[43] Yanlan Ding and Guiping Su. Intrusion detection system for signal based SIP attacks through timed HCPN. In *Availability, Reliability and Security, 2007. ARES 2007. The Second International Conference on*, pages 190–197. IEEE, 2007.

[44] Stelios Dritsas, Vicky Dritsou, Bill Tsoumas, Panos Constantopoulos, and Dimitris Gritzalis. OntoSPIT:SPIT management through ontologies. *Computer Communications*, 32(1):203–212, 2009.

[45] Sven Ehlert, Dimitris Geneiatakis, and Thomas Magedanz. Survey of network security systems to counter SIP-based denial-of-service attacks. *Computers & Security*, 29(2):225–243, 2010.

[46] Sven Ehlert, Chengjian Wang, Thomas Magedanz, and Dorgham Sisalem. Specification-based denial-of-service detection for SIP Voice over IP networks. In *Internet Monitoring and Protection, 2008. ICIMP'08. The Third International Conference on*, pages 59–66. IEEE, 2008.

[47] Sven Ehlert, Ge Zhang, Dimitris Geneiatakis, Georgios Kambourakis, Tasos Dagiuklas, Jiří Markl, and Dorgham Sisalem. Two layer Denial of Service prevention on SIP VoIP infrastructures. *Computer Communications*, 31(10):2443–2456, 2008.

[48] David Endler, Mark Collier, et al. *Hacking Exposed VoIP.* Tata McGraw-Hill Education, 2007.

[49] Antonio Estepa, Rafael Estepa, and J Vozmediano. A new approach for VoIP traffic characterization. *Communications Letters, IEEE*, 8(10):644–646, 2004.

[50] Pablo A Estévez, Claudio M Held, and Claudio A Perez. Subscription fraud prevention in telecommunications using fuzzy rules and neural networks. *Expert Systems with Applications*, 31(2):337–344, 2006.

[51] Marco Falomi, R Garroppo, and Saverio Niccolini. Simulation and optimization of SPIT detection frameworks. In *Global Telecommunications Conference, 2007. GLOBECOM'07. IEEE*, pages 2156–2161. IEEE, 2007.

[52] Hamid Farvaresh and Mohammad Mehdi Sepehri. A data mining framework for detecting subscription fraud in telecommunication. *Engineering Applications of Artificial Intelligence*, 24(1):182–194, 2011.

[53] Dieter Fensel. *Ontologies.* Springer, 2001.

[54] Dieter Fensel, Frank Van Harmelen, Michel Klein, Hans Akkermans, Jeen Broekstra, Christiaan Fluit, Jos van der Meer, Hans-Peter Schnurr, Rudi Studer, John Hughes, et al. On-to-knowledge: Ontology-based tools for knowledge management. In *Proceedings of the eBusiness and eWork*, pages 18–20, 2000.

[55] Raihana Ferdous, Renato Lo Cigno, and Alessandro Zorat. Social Behavior Analysis of VoIP Users and its application to Malicious Users Detection (Extended Version V1. 0). 2014.

[56] Raihana Ferdous, Renato Lo Cigno, and Alessandro Zorat. Classification of SIP messages by a syntax filter and SVMs. In *IEEE Global Communications Conference 2012 (GLOBECOM'12)*, pages 2714–2719, Anheim, CA, Dec. 3-7, 2012.

[57] Raihana Ferdous, Renato Lo Cigno, and Alessandro Zorat. On the Use of SVMs to Detect Anomalies in a Stream of SIP Messages. In *11th IEEE Int. Conference on Machine Learning and Applications (ICMLA)*, pages 592–597, Boca Raton, FL, Dec. 12-15, 2012.

[58] Chris Fraley and Adrian E Raftery. How many clusters? which clustering method? answers via model-based cluster analysis. *The computer journal*, 41(8):578–588, 1998.

[59] Chris Fraley, Adrian E Raftery, T Brendan Murphy, and Luca Scrucca. MCLUST version 4 for R: normal mixture modeling for model-based clustering, classification, and density estimation. Technical report, 2012.

[60] John Franks, Phillip Hallam-Baker, Jeffrey Hostetler, Scott Lawrence, Paul Leach, Ari Luotonen, and Lawrence Stewart. HTTP Authentication: Basic and Digest Access Authentication, 1999.

[61] M Garcia-Martin. Input 3rd-generation partnership project (3GPP) release 5 requirements on the Session Initiation Protocol (SIP). 2005.

[62] Pedro Garcia-Teodoro, J Diaz-Verdejo, Gabriel Maciá-Fernández, and Enrique Vázquez. Anomaly-based network intrusion detection: Techniques, systems and challenges. *computers & security*, 28(1):18–28, 2009.

[63] Dimitris Geneiatakis, Tasos Dagiuklas, Georgios Kambourakis, Costas Lambrinoudakis, Stefanos Gritzalis, Sven Ehlert, Dorgham Sisalem, et al. Survey of security vulnerabilities in Session Initiation Protocol. *IEEE Communications Surveys and Tutorials*, 8(1-4):68–81, 2006.

[64] Dimitris Geneiatakis, Georgios 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.

[65] Dimitris Geneiatakis, Georgios Kambourakis, Costas Lambrinoudakis, Tasos Dagiuklas, and Stefanos Gritzalis. A framework for protecting a SIP-based infrastructure against malformed message attacks. *Comput. Netw.*, 51(10):2580–2593, jul 2007.

[66] Dimitris Geneiatakis and Costas Lambrinoudakis. An ontology description for SIP security flaws. *Computer Communications*, 30(6):1367–1374, 2007.

[67] Dimitris Geneiatakis, Costas Lambrinoudakis, and Georgios Kambourakis. An ontology-based policy for deploying secure SIP-based VoIP services. *Computers & Security*, 27(7):285–297, 2008.

[68] Dimitris Geneiatakis, Nikos Vrakas, and Costas Lambrinoudakis. Utilizing bloom filters for detecting flooding attacks against SIP based services. *computers & security*, 28(7):578–591, 2009.

[69] Michelle Girvan and Mark EJ Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, 2002.

[70] Dimitris Gritzalis, Panagiotis Katsaros, Stylianos Basagiannis, and Yannis Soupionis. Formal analysis for robust anti-SPIT protection using model checking. *International Journal of Information Security*, 11(2):121–135, 2012.

[71] Dimitris Gritzalis and Yannis Mallios. A SIP-oriented SPIT management framework. *Computers & Security*, 27(5):136–153, 2008.

[72] Markus Gruber, Christian Schanes, Florian Fankhauser, Martin Moutran, and Thomas Grechenig. Architecture for Trapping Toll Fraud Attacks Using a VoIP Honeynet Approach. In *Network and System Security*, pages 628–634. Springer, 2013.

[73] Thomas R Gruber et al. A translation approach to portable ontology specifications. *Knowledge acquisition*, 5(2):199–220, 1993.

[74] He Guang-Yu, Wen Ying-You, and Zhao Hong. SPIT detection and prevention method in VoIP environment. In *Availability, Reliability and Security, 2008. ARES 08. Third International Conference on*, pages 473–478. IEEE, 2008.

[75] D. Gusfield. *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology.* Cambridge University Press, 1997.

[76] Do-Yoon Ha, Hwan-Kuk Kim, Kyoung-Hee Ko, Chang-Yong Lee, Jeong-Wook Kim, and Hyun-Cheol Jeong. Design and implementation of SIP-aware DDoS attack detection system. In *Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human*, pages 1167–1171. ACM, 2009.

[77] Ehsan Haghani, Swades De, and Nirwan Ansari. On modeling VoIP traffic in broadband networks. In *Global Telecommunications Conference, 2007. GLOBECOM'07. IEEE*, pages 1922–1926. IEEE, 2007.

[78] John A Hartigan and Manchek A Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108, 1979.

[79] H Hassan, JM Garcia, and C Bockstal. Aggregate traffic models for VoIP applications. In *Digital Telecommunications,, 2006. ICDT'06. International Conference on*, pages 70–70. IEEE, 2006.

[80] Poul E Heegaard. Empirical observations of traffic patterns in mobile and ip telephony. In *Next Generation Teletraffic and Wired/Wireless Advanced Networking*, pages 26–37. Springer, 2007.

[81] N. Hentehzadeh, A. Mehta, V.K. Gurbani, L. Gupta, Tin Kam Ho, and G. Wilathgamuwa. Statistical analysis of self-similar Session Initiation Protocol (SIP) Messages for

Anomaly Detection. In *4th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*, pages 1 –5, feb. 2011.

[82] Almut Herzog, Nahid Shahmehri, and Claudiu Duma. An ontology of information security. *International Journal of Information Security and Privacy (IJISP)*, 1(4):1–23, 2007.

[83] Constantinos S Hilas. Designing an expert system for fraud detection in private telecommunications networks. *Expert Systems with applications*, 36(9):11559–11569, 2009.

[84] Constantinos S Hilas and Paris As Mastorocostas. An application of supervised and unsupervised learning approaches to telecommunications fraud detection. *Knowledge-Based Systems*, 21(7):721–726, 2008.

[85] Stefan Hofbauer, Kristian Beckers, Gerald Quirchmayr, and Christoph Sorge. A lightweight privacy preserving approach for analyzing communication records to prevent VoIP attacks using toll fraud as an example. In *Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on*, pages 992–997. IEEE, 2012.

[86] Dirk Hoffstadt, Alexander Marold, and Erwin P Rathgeb. Analysis of SIP-based threats using a VoIP honeynet system. In *Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on*, pages 541–548. IEEE, 2012.

[87] ChihWei Hsu, ChihChung Chang, and ChihJen Lin. A practical guide to support vector classification. Technical report, Department of Computer Science, National Taiwan University, 2003.

[88] Felipe Huici, Saverio Niccolini, and Nico d'Heureuse. Protecting SIP against very large flooding dos attacks. In *Global Telecommunications Conference, 2009. GLOBECOM 2009. IEEE*, pages 1–6. IEEE, 2009.

[89] Philip Hunter. VoIP the latest security concern: DoS attack the greatest threat. *Network Security*, 2002(11):5–7, 2002.

[90] Seyed Amir Iranmanesh, Hemant Sengar, and Haining Wang. A Voice Spam Filter to Clean Subscribers Mailbox. In *Security and Privacy in Communication Networks*, pages 349–367. Springer, 2013.

[91] Cullen Jennings. Computational Puzzles for SPAM Reduction in SIP. 2007.

[92] Kurt Jensen. *Coloured petri nets.* Springer, 1987.

[93] Li Ji, Xia Yin, Xingang Shi, and Zhiliang Wang. Conversational model based VoIP traffic generation. In *Networking and Services, 2007. ICNS. Third International Conference on*, pages 14–14. IEEE, 2007.

[94] I. T. Jolliffe. *Principal Component Analysis*. Springer, second edition, oct 2002.

[95] Tobias Jung, Sylvain Martin, Mohamed Nassar, Damien Ernst, and Guy Leduc. Outbound SPIT filter with optimal performance guarantees. *Computer Networks*, 2013.

[96] Igor Jurisica, John Mylopoulos, and Eric Yu. Ontologies for knowledge management: an information systems perspective. *Knowledge and Information systems*, 6(4):380–401, 2004.

[97] Hun Jeong Kang, Zhi-Li Zhang, Supranamaya Ranjan, and Antonio Nucci. SIP-based VoIP traffic behavior profiling and its applications. In *Proceedings of the 3rd annual ACM workshop on Mining network data*, pages 39–44. ACM, 2007.

[98] Tapas Kanungo, David M Mount, Nathan S Netanyahu, Christine D Piatko, Ruth Silverman, and Angela Y Wu. An efficient k-means clustering algorithm: Analysis and implementation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(7):881–892, 2002.

[99] Rezaul Karim, Martin Bergtholdt, Jrg H. Kappes, and Christoph Schnrr. Greedy-based design of sparse two-stage svms for fast classification. In Fred A. Hamprecht, Christoph Schnrr, and Bernd Jhne, editors, *DAGM-Symposium*, volume 4713 of *Lecture Notes in Computer Science*, pages 395–404. Springer, 2007.

[100] Stephen Kent and Randall Atkinson. Security architecture for the internet protocol, 1998.

[101] A.D. Keromytis. *Voice Over IP Security: A Comprehensive Survey of Vulnerabilities and Academic Research*. Springerbriefs in Computer Science. Springer, 2011.

[102] Angelos D Keromytis. Voice over IP: Risks, Threats and Vulnerabilities. *Cyber Infrastructure Protection*, 2009.

[103] David J. Ketchen and Christopher L. Shook. The Application of Cluster Analysis in Strategic Management Research: An Analysis and Critique. *Strategic Management Journal*, 17(6):441–458, 1996.

[104] Saeed Farooq Khan, Marius Portmann, and Neil W Bergmann. VoIP Spam Prevention. In *Trust, Security and Privacy in Computing and Communications (TrustCom), 2013 12th IEEE International Conference on*, pages 1463–1470. IEEE, 2013.

[105] Dong Seong Kim and Jong Sou Park. Network-based intrusion detection with support vector machines. In *Information Networking*, pages 747–756. Springer, 2003.

[106] Hyung-Jong Kim, Myuhng Joo Kim, Yoonjeong Kim, and Hyun Cheol Jeong. DEVS-Based modeling of VoIP spam callers behavior for SPIT level calculation. *Simulation Modelling Practice and Theory*, 17(4):569–584, 2009.

[107] Takahisa Kitagawa, Miyuki Hanaoka, and Kenji Kono. AspFuzz: A state-aware protocol fuzzer based on application-layer protocols. In *Computers and Communications (ISCC), 2010 IEEE Symposium on*, pages 202–208. IEEE, 2010.

[108] Prakash Kolan and Ram Dantu. Socio-technical defense against voice spamming. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 2(1):2, 2007.

[109] D Richard Kuhn, Thomas J Walsh, and Steffen Fries. Security considerations for voice over ip systems. *NIST special publication*, pages 800–58, 2005.

[110] Tetsuya Kusumoto, Eric Y Chen, and Mitsutaka Itoh. Using call patterns to detect unwanted communication callers. In *Applications and the Internet, 2009. SAINT'09. Ninth Annual International Symposium on*, pages 64–70. IEEE, 2009.

[111] Andrea Lancichinetti and Santo Fortunato. Community detection algorithms: a comparative analysis. *Physical review E*, 80(5):056117, 2009.

[112] Aleksandar Lazarevic, Levent Ertöz, Vipin Kumar, Aysel Ozgur, and Jaideep Srivastava. A Comparative Study of Anomaly Detection Schemes in Network Intrusion Detection. In *SDM*. SIAM, 2003.

[113] Hongbin Li, Hu Lin, Xuehua Yang, and Feng Liu. A rules-based intrusion detection and prevention framework against SIP malformed messages attacks. In *3rd IEEE International Conference on Broadband Network and Multimedia Technology (IC-BNMT)*, pages 700 – 705, oct. 2010.

[114] Wan Li and Shengfeng Tian. An ontology-based intrusion alerts correlation system. *Expert Systems with Applications*, 37(10):7138–7146, 2010.

[115] XiaoOu Li, J. Cervantes, and Wen Yu. Two-stage svm classification for large data sets via randomly reducing and recovering training data. In *Systems, Man and Cybernetics, 2007. ISIC. IEEE International Conference on*, pages 3633–3638, 2007.

[116] Lin Liu. Uncovering SIP vulnerabilities to dos attacks using coloured petri nets. In *Trust, Security and Privacy in Computing and Communications (TrustCom), 2011 IEEE 10th International Conference on*, pages 29–36. IEEE, 2011.

[117] Robert MacIntosh and Dmitri Vinokurov. Detection and mitigation of spam in IP telephony networks using signaling protocol analysis. In *Advances in Wired and Wireless Communication, 2005 IEEE/Sarnoff Symposium on*, pages 49–52. IEEE, 2005.

[118] Giannis F Marias, Stelios Dritsas, Marianthi Theoharidou, John Mallios, and Dimitris Gritzalis. SIP vulnerabilities and anti-SPIT mechanisms assessment. In *Computer Communications and Networks, 2007. ICCCN 2007. Proceedings of 16th International Conference on*, pages 597–604. IEEE, 2007.

[119] Bertrand Mathieu, Saverio Niccolini, and Dorgham Sisalem. SDRS: A Voice-over-IP Spam Detection and Reaction System. *Security & Privacy, IEEE*, 6(6):52–59, 2008.

[120] Peter M. Maurer. Generating test data with enhanced context-free grammars. *Software, IEEE*, 7(4):50–55, 1990.

[121] Shawn McGann and Douglas C Sicker. An analysis of security threats and tools in SIP-based VoIP systems. In *Second VoIP security workshop*, 2005.

[122] Federico Menna, R Lo Cigno, Saverio Niccolini, and Sandra Tartarelli. Simulation of SPIT filtering: Quantitative evaluation of parameter tuning. In *Communications, 2009. ICC'09. IEEE International Conference on*, pages 1–6. IEEE, 2009.

[123] Yves Moreau, Herman Verrelst, and Joos Vandewalle. Detection of mobile phone fraud using supervised neural networks: A first prototype. In *Artificial Neural NetworksICANN'97*, pages 1065–1070. Springer, 1997.

[124] Biswanath Mukherjee, L Todd Heberlein, and Karl N Levitt. Network intrusion detection. *Network, IEEE*, 8(3):26–41, 1994.

[125] Srinivas Mukkamala, Guadalupe Janoski, and Andrew Sung. Intrusion detection using neural networks and support vector machines. In *Neural Networks, 2002. IJCNN'02. Proceedings of the 2002 International Joint Conference on*, volume 2, pages 1702–1707. IEEE, 2002.

[126] Mohamed Nassar, Sylvain Martin, Guy Leduc, and Olivier Festor. Using decision trees for generating adaptive SPIT signatures. In *Proceedings of the 4th international conference on Security of information and networks*, pages 13–20. ACM, 2011.

[127] Mohamed Nassar, Saverio Niccolini, Thilo Ewald, et al. Holistic VoIP intrusion detection and prevention system. In *Proceedings of the 1st international conference on Principles, systems and applications of IP telecommunications*, pages 1–9. ACM, 2007.

[128] Mohamed Nassar, R State, and Olivier Festor. VoIP malware: Attack tool & attack scenarios. In *Communications, 2009. ICC'09. IEEE International Conference on*, pages 1–6. IEEE, 2009.

[129] Mohamed Nassar, Radu State, and Olivier Festor. Monitoring SIP Traffic Using Support Vector Machines. In *Recent Advances in Intrusion Detection (RAID '08)*, volume 5230, pages 311–330. Springer Berlin / Heidelberg, 2008.

[130] Mark EJ Newman and Michelle Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.

[131] S. Niccolini. VoIP Security Threats. Internet-Draft.

[132] MS Nikulin. Hellinger distance. *Encyclopaedia of Mathematics. Kluwer Academic Publishers*, 2002.

[133] Stephen Northcutt and Judy Novak. *Network intrusion detection.* Sams Publishing, 2002.

[134] Dominik Olszewski. A probabilistic approach to fraud detection in telecommunications. *Knowledge-Based Systems*, 26:246–258, 2012.

[135] Jukka-Pekka Onnela, Jari Saramäki, Jörkki Hyvönen, Gábor Szabó, M Argollo De Menezes, Kimmo Kaski, Albert-László Barabási, and János Kertész. Analysis of a large-scale weighted network of one-to-one human communication. *New Journal of Physics*, 9(6):179, 2007.

[136] Kumiko Ono and Henning Schulzrinne. Have I met you before?: using cross-media relations to reduce SPIT. In *Proceedings of the 3rd International Conference on Principles, Systems and Applications of IP Telecommunications*, page 3. ACM, 2009.

[137] Gaston Ormazabal, Sarvesh Nagpal, Eilon Yardeni, and Henning Schulzrinne. Secure SIP: A scalable prevention mechanism for dos attacks on SIP based VoIP systems. In *Principles, systems and applications of IP telecommunications. Services and security for next generation networks*, pages 107–132. Springer, 2008.

[138] Bijan Parsia and Evren Sirin. Pellet: An owl dl reasoner. In *Third International Semantic Web Conference-Poster*, volume 18, 2004.

[139] Jennings Peterson, C Jennings, et al. Enhancements for authenticated identity management in the Session Initiation Protocol (SIP). Technical report, RFC 4474, August, 2006.

[140] Carlos Andre Reis Pinheiro. *Social network analysis in telecommunications*, volume 37. John Wiley & Sons, 2011.

[141] Craig Pollard. Telecom fraud: The cost of doing nothing just went up. *Computers & Security*, 24(6):437–439, 2005.

[142] Mason A Porter, Jukka-Pekka Onnela, and Peter J Mucha. Communities in networks. *Notices of the AMS*, 56(9):1082–1097, 2009.

[143] Jürgen Quittek, Saverio Niccolini, Sandra Tartarelli, Martin Stiemerling, Marcus Brunner, and Thilo Ewald. Detecting SPIT calls by checking human communication patterns. In *Communications, 2007. ICC'07. IEEE International Conference on*, pages 1979–1984. IEEE, 2007.

[144] 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 *Eighth International Conference on Information Technology: New Generations (ITNG)*, pages 559 –563, april 2011.

[145] Blake Ramsdell. Secure/multipurpose internet mail extensions (S/MIME) version 3.1 message specification. (RFC 3851), 2004.

[146] Victor Raskin, Christian F Hempelmann, Katrina E Triezenberg, and Sergei Nirenburg. Ontology in information security: a useful theoretical foundation and methodological tool. In *Proceedings of the 2001 workshop on New security paradigms*, pages 53–59. ACM, 2001.

[147] Abdul Razzaq, HF Ahmed, A Hur, and Nasir Haider. Ontology based application level intrusion detection system by using bayesian filter. In *Computer, Control and Communication, 2009. IC4 2009. 2nd International Conference on*, pages 1–6. IEEE, 2009.

[148] Yacine Rebahi, Mohamed Nassar, Thomas Magedanz, and Olivier Festor. A survey on fraud and service misuse in Voice over IP (VoIP) networks. *Information Security Technical Report*, 16(1):12–19, 2011.

[149] Yacine Rebahi and Dorgham Sisalem. Change-point detection for Voice over IP denial of service attacks. In *Communication in Distributed Systems (KiVS), 2007 ITG-GI Conference*, pages 1–7. VDE, 2007.

[150] Yacine Rebahi, Dorgham Sisalem, and Thomas Magedanz. SIP Spam Detection. In *Digital Telecommunications, 2006. ICDT'06. International Conference on*, pages 68–68. IEEE, 2006.

[151] Brennen Reynolds and Dipak Ghosal. Secure IP Telephony using Multi-layered Protection. In *NDSS*, 2003.

[152] Konrad Rieck, Stefan Wahl, Pavel Laskov, Peter Domschitz, and Klaus-Robert Müller. A Self-learning System for Detection of Anomalous SIP Messages. In *Principles, Systems and Applications of IP Telecommunications. Services and Security for Next Generation Networks*, volume 5310, pages 90–106. Springer Berlin / Heidelberg, 2008.

[153] Ronald Rivest. The MD5 message-digest algorithm. 1992.

[154] Martin Roesch et al. Snort: Lightweight Intrusion Detection for Networks. In *LISA*, volume 99, pages 229–238, 1999.

[155] J Rosenberg, C Jennings, and J Peterson. The Session Initiation Protocol (SIP) and spam. Technical report, RFC 5039, January, 2008.

[156] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: Session Initiation Protocol. RFC 3261, June 2002.

[157] Saharon Rosset, Uzi Murad, Einat Neumann, Yizhak Idan, and Gadi Pinkas. Discovery of fraud rules for telecommunicationschallenges and solutions. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 409–413. ACM, 1999.

[158] Igor Ruiz-Agundez, Yoseba K Penya, and Pablo Garcia Bringas. Fraud detection for Voice over IP services on next-generation networks. In *Information Security Theory and Practices. Security and Privacy of Pervasive Systems and Smart Devices*, pages 199–212. Springer, 2010.

[159] S. Salvador and P. Chan. Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms. In *Tools with Artificial Intelligence, 2004. ICTAI 2004. 16th IEEE International Conference on*, pages 576–584, 2004.

[160] Roman Schlegel, Saverio Niccolini, Sandra Tartarelli, and Marcus Brunner. Spam over Internet Telephony (SPIT) prevention framework. In *Global Telecommunications Conference, 2006. GLOBECOM'06. IEEE*, pages 1–6. IEEE, 2006.

[161] Henning Schulzrinne. RTP: A transport protocol for real-time applications. (RFC 1889), 1996.

[162] Gideon Schwarz. Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464, 1978.

[163] Hemant Sengar, Haining Wang, Duminda Wijesekera, and Sushil Jajodia. Fast detection of denial-of-service attacks on IP telephony. In *Quality of Service, 2006. IWQoS 2006. 14th IEEE International Workshop on*, pages 199–208. IEEE, 2006.

[164] Hemant Sengar, Haining Wang, Duminda Wijesekera, and Sushil Jajodia. Detecting VoIP floods using the Hellinger distance. *Parallel and Distributed Systems, IEEE Transactions on*, 19(6):794–805, 2008.

[165] Hemant Sengar, Xinyuan Wang, and Art Nichols. Thwarting spam over internet telephony (SPIT) attacks on VoIP networks. In *Proceedings of the Nineteenth International Workshop on Quality of Service*, page 25. IEEE Press, 2011.

[166] Hemant Sengar, Xinyuan Wang, and Arthur Nichols. Call Behavioral Analysis to Thwart SPIT Attacks on VoIP Networks. In *Security and Privacy in Communication Networks*, pages 501–510. Springer, 2012.

[167] Hemant Sengar, Duminda Wijesekera, Haining Wang, and Sushil Jajodia. VoIP intrusion detection through interacting protocol state machines. In *Dependable Systems and Networks, 2006. DSN 2006. International Conference on*, pages 393–402. IEEE, 2006.

[168] Dongwon Seo, Heejo Lee, and Ejovi Nuwere. Detecting more SIP attacks on VoIP services by Combining Rule Matching and State Transition Models. In *Proceedings of The International Federation for Information Processing (IFIP)*, volume 278, pages 397–411. Springer Boston, 2008.

[169] Mukund Seshadri, Sridhar Machiraju, Ashwin Sridharan, Jean Bolot, Christos Faloutsos, and Jure Leskove. Mobile call graphs: beyond power-law and lognormal distributions. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 596–604. ACM, 2008.

[170] John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis.* Cambridge University Press, Cambridge, UK, 2004.

[171] Dongwook Shin, Jinyoung Ahn, and Choon Shim. Progressive multi gray-leveling: a voice spam protection algorithm. *Network, IEEE*, 20(5):18–24, 2006.

[172] "SIPp". A free open source test tool/traffic generator for the sip protocol. `http://sipp.sourceforge.net/`.

[173] Dorgham Sisalem, Jiri Kuthan, and Sven Ehlert. Denial of service attacks targeting a SIP VoIP infrastructure: attack scenarios and prevention mechanisms. *Network, IEEE*, 20(5):26–31, 2006.

[174] So Young Sohn and Yoonseong Kim. Searching customer patterns of mobile service using clustering and quantitative association rule. *Expert Systems with Applications*, 34(2):1070–1077, 2008.

[175] Yannis Soupionis, Stylianos Basagiannis, Panagiotis Katsaros, and Dimitris Gritzalis. A formally verified mechanism for countering SPIT. In *Critical Information Infrastructures Security*, pages 128–139. Springer, 2011.

[176] Yannis Soupionis and Dimitris Gritzalis. Audio CAPTCHA: Existing solutions assessment and a new implementation for VoIP telephony. *Computers & Security*, 29(5):603–618, 2010.

[177] Robert J Sparks, Jonathan Rosenberg, Henning Schulzrinne, Alan Hawrylyshen, and Alan Johnston. Session Initiation Protocol (SIP) torture test messages. 2006.

[178] Hemanth Srinivasan and Kamil Sarac. A SIP security testing framework. In *Consumer Communications and Networking Conference, 2009. CCNC 2009. 6th IEEE*, pages 1–5. IEEE, 2009.

[179] Jan Stanek and Lukas Kencl. SIP Protector: Defense architecture mitigating DDoS flood attacks against SIP servers. In *Communications (ICC), 2012 IEEE International Conference on*, pages 6733–6738. IEEE, 2012.

[180] Julian Strobl, Bernhard Mainka, Gary Grutzek, and Heiko Knospe. An efficient search method for the content-based identification of telephone-SPAM. In *Communications (ICC), 2012 IEEE International Conference on*, pages 2623–2627. IEEE, 2012.

[181] Wai Man Tam, Francis Chung-Ming Lau, and CK Tse. Modeling the telephone call network. In *Circuits and Systems, 2007. ISCAS 2007. IEEE International Symposium on*, pages 453–456. IEEE, 2007.

[182] Jin Tang, Yu Cheng, and Chi Zhou. Sketch-based SIP flooding detection using Hellinger distance. In *Global Telecommunications Conference, 2009. GLOBECOM 2009. IEEE*, pages 1–6. IEEE, 2009.

[183] Michiaki Taniguchi, Michael Haft, Jaakko Hollmén, and Volker Tresp. Fraud detection in communication networks using neural and probabilistic methods. In *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, volume 2, pages 1241–1244. IEEE, 1998.

[184] P. Thermos and A. Takanen. *Securing VoIP networks: threats, vulnerabilities, and countermeasures*. Addison-Wesley, 2008.

[185] CISSP Thomas Porter, CCDA CCNP, et al. *Practical VoIP Security*. Syngress, 2006.

[186] RobertL. Thorndike. Who belongs in the family? *Psychometrika*, 18(4):267–276, 1953.

[187] Marina Thottan and Chuanyi Ji. Anomaly detection in IP networks. *Signal Processing, IEEE Transactions on*, 51(8):2191–2204, 2003.

[188] Kentaroh Toyoda and Iwao Sasase. SPIT callers detection with unsupervised random forests classifier. In *Communications (ICC), 2013 IEEE International Conference on*, pages 2068–2072. IEEE, 2013.

[189] Jeffrey Undercoffer, John Pinkston, Anupam Joshi, and Timothy Finin. A target-centric ontology for intrusion detection. In *18th International Joint Conference on Artificial Intelligence*, pages 9–15, 2004.

[190] RP Van Heerden, B Irwin, and ID Burke. Classifying network attack scenarios using an ontology. Academic Conferences Limited, 2012.

[191] Jim Van Meggelen, Leif Madsen, and Jared Smith. *Asterisk: the future of telephony.* " O'Reilly Media, Inc.", 2007.

[192] Vladimir N. Vapnik. *The nature of statistical learning theory.* Springer-Verlag New York, Inc., NY, USA, 1995.

[193] France Villers-Les-Nancy. Labeled VoIP Data-Set for Intrusion Detection Evaluation. *Networked Services and Applications–Engineering, Control and Management*, page 97, 2010.

[194] TK Vintsyuk. Speech discrimination by dynamic programming. *Cybernetics and Systems Analysis*, 4(1):52–57, 1968.

[195] Luis Von Ahn, Manuel Blum, and John Langford. Telling humans and computers apart automatically. *Communications of the ACM*, 47(2):56–60, 2004.

[196] Artem Vorobiev and Nargiza Bekmamedova. An ontology-driven approach applied to information security. *Journal of Research and Practice in Information Technology*, 42(1):61–76, 2010.

[197] Dashun Wang, Dino Pedreschi, Chaoming Song, Fosca Giannotti, and Albert-Laszlo Barabasi. Human mobility, social ties, and link prediction. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1100–1108. ACM, 2011.

[198] Fei Wang, Min Feng, and KeXing Yan. Voice Spam Detecting Technique Based on User Behavior Pattern Model. In *Wireless Communications, Networking and Mobile Computing (WiCOM), 2012 8th International Conference on*, pages 1–5. IEEE, 2012.

[199] Xinyuan Wang, Ruishan Zhang, Xiaohui Yang, Xuxian Jiang, and Duminda Wijesekera. Voice pharming attack and the trust of VoIP. In *Proceedings of the 4th international conference on Security and privacy in communication netowrks*, page 24. ACM, 2008.

[200] Stanley Wasserman. Social Network Analysis: Methods and Applications. 1994.

[201] Christian Wieser, Marko Laakso, and Henning G Schulzrinne. Security testing of SIP implementations. 2003. Web – `https://www.ee.oulu.fi/research/ouspg/PROTOS_Test-Suite_c07-sip`.

[202] Yu-Sung Wu, Saurabh Bagchi, Sachin Garg, and Navjot Singh. SCIDIVE: a stateful and cross protocol intrusion detection architecture for Voice-over-IP environments. In

Dependable Systems and Networks, 2004 International Conference on, pages 433–442. IEEE, 2004.

[203] Yu-Sung Wu, Saurabh Bagchi, Navjot Singh, and Ratsameetip Wita. Spam detection in Voice-over-IP calls through semi-supervised clustering. In *Dependable Systems & Networks, 2009. DSN'09. IEEE/IFIP International Conference on*, pages 307–316. IEEE, 2009.

[204] Dongshan Xing and Mark Girolami. Employing Latent Dirichlet Allocation for fraud detection in telecommunications. *Pattern Recognition Letters*, 28(13):1727–1734, 2007.

[205] Hong Yan, Kunwadee Sripanidkulchai, Hui Zhang, Zon-Yin Shae, and Debanjan Saha. Incorporating active fingerprinting into SPIT prevention systems. In *Third annual security workshop (VSW06)*, 2006.

[206] Jonathan Zar et al. VOIPSA VoIP Security and Privacy Threat Taxonomy-Public release 0.1. Technical report, 2005.

[207] Ge Zhang, Sven Ehlert, Thomas Magedanz, and Dorgham Sisalem. Denial of service attack and prevention on SIP VoIP infrastructures using DNS flooding. In *Proceedings of the 1st international conference on Principles, systems and applications of IP telecommunications*, pages 57–66. ACM, 2007.

[208] Hongli Zhang, Zhimin Gu, Caixia Liu, and Tang Jie. Detecting VoIP-specific denial-of-service using change-point method. In *Advanced Communication Technology, 2009. ICACT 2009. 11th International Conference on*, volume 2, pages 1059–1064. IEEE, 2009.

[209] Ruishan Zhang and Andrei Gurtov. Collaborative reputation-based voice spam filtering. In *Database and Expert Systems Application, 2009. DEXA'09. 20th International Workshop on*, pages 33–37. IEEE, 2009.

[210] Ruishan Zhang, Xinyuan Wang, Ryan Farley, Xiaohui Yang, and Xuxian Jiang. On the feasibility of launching the man-in-the-middle attacks on VoIP from remote attackers. In *Proceedings of the 4th International Symposium on Information, Computer, and Communications Security*, pages 61–69. ACM, 2009.

Appendix A

List of Features

According to the architecture of our SIP message filtering system described in Chapter 6, after the first stage checking for *syntax* error, each well-formed message is further processed where 26 significant features extracted from a SIP message. These features are selected by expert knowledge about VoIP systems and they take into consideration aspects related to the structure and the content of a SIP message.

Table A.1: List of Features for classification of SIP messages

ID	Feature	Description	Value Range
1	Message Type	This feature indicates the type (Request/Response) of a SIP message	0=Unknown Method, 1-14=Request Method, 15=Response Message
2	Request Line Status	This feature indicates the status and frequency (absence/multiple) of Request-Line in a Request Message.It is required to identify error in Request-Line	-1= In wrong position, 0 = No Request-Line, 1=In perfect position , >1=Multiple Request-Line
3	Protocol Version	This feature contains information about SIP protocol version. It is required to detect error, such as, unknown or higher protocol version	SIP protocol version

Continued on next page

Table A.1 – continued from previous page

ID	Feature	Description	Value Range
4	Empty Line Status	This feature indicates the status and occurrence of empty line in a message. It is required to detect hierarchical disorder in a message due to multiple/incorrect presence of empty line in a message	-1=in wrong position, 0=no empty line, >1=multiple empty line
5	Message order	This feature contains information about the hierarchical order of a SIP message. It holds information of any kind of disorder in a message	1=message in order, -1=message is not in order
6	Presence of garbage string	This feature indicates the presence of garbage strings in a message	-1=presence of garbage string, 1=no garbage string
7	Response Line Status	This feature indicates the status (absence/multiple) of Response-Line in a Response Message	Frequency of Request-line in a message
8	Scalar value of "CSeq" header	This feature contains the value of the scalar field in 'CSeq' header	Value of scalar field in "CSeq" header
9	Scalar value of "Max-Forwards" header	This feature contains the value of the scalar field in "Max-Forwards" header	Value of scalar field in 'Max-Forwards' header
10	Scalar value of "Content-Length" header	This feature contains the value of the scalar field in "Content-Length" header	Value of scalar field in 'Content-Length' header
11	Missing Mandatory Header	This feature contains the information whether any specific mandatory header field is missing in a message.	0=All mandatory fields are present, -1=missing mandatory header fields in Response message
12	Method Name	This feature indicates the method name of a Request message	0=Unknown Method Name, 1–14=Valid Request Method
13	Request-URI status	This feature contains the status of Request-URI. This feature is used to identify unknown Request-URI scheme in a Request message	-1=Unknown Request-URI scheme, 1=Correct Request-URI scheme
14	IP address in "Via" header	This feature contains the IP address of "Via" header field of a Response message. This feature is required to identify Response message which intends to broadcast due to IP ("255.255.255.255") in 'Via' header	1= header ok, -1=error in IP address

Continued on next page

Table A.1 – continued from previous page

ID	Feature	Description	Value Range
15	Size of Response code	This feature indicates the length of a Response Message. It is used to detect very large Response message	Length of the Response Message
16	Mandatory header field "Call-ID" status	This feature indicates the occurrence of header field "Call-ID" in a message	Frequency of header field 'Call-ID' in a message. 0= Missing, 1=Single appearance, 2= Duplicate appearance... of "Call-ID" in a message
17	Mandatory header field "CSeq" status	This feature indicates the occurrence of header field "CSeq" in a message	Frequency of header field "CSeq" in a message. 0= Missing, 1= Single appearance and ok, >1= Multiple appearances
18	Unknown Header Field	This feature indicates the presence of unknown header field in a message.	1=no unknown header field, -1=unknown header field
19	Mandatory header field "Contact" status	This feature indicates the occurrence of header field "Contact" in a message. It is used to detect error such as missing or multiple occurrence of "Contact" header field in a message.	Frequency of header field "Contact" in a message. 0=Missing, 1=Single appearance, >1=Multiple appearances
20	Mandatory header field "From" status	This feature indicates the occurrence of header field "From" in a message. It is used to detect error such as missing or multiple occurrence of "From" header field in a message.	Frequency of header field "From" in a message. 0= Missing, 1=Single appearance, >1 =Multiple appearances
21	Mandatory header field "Max-Forwards" status	This feature indicates the occurrence of header field "Max-Forwards" in a message. It is used to detect error such as missing or multiple occurrence of "Max-Forwards" header field in a message.	Frequency of header field "Max-Forwards" in a message. 0=Missing, 1=Single appearance, >1=Multiple appearances
22	Mandatory header field "To" status	This feature indicates the occurrence of header field "To" in a message. It is used to detect error such as missing or multiple occurrence of "To" header field in a message	Frequency of header field "To" in a message. 0= Missing, 1=Single appearance, >1=Multiple appearances

Continued on next page

Table A.1 – continued from previous page

ID	Feature	Description	Value Range
23	Frequency header field "Via" status	This feature indicates the occurrence of header field "Via" in a message. It is used to detect error such as missing or multiple occurrence of "Via" header field in a message	Frequency of header field "Via" in a message. 0=Missing, 1=Single appearance, >1=Multiple appearances
24	"Authentication_Info" header field status	This feature contains information about the "Authentication_Info" header fields. This feature is used to detect any error in this field, such as unauthorized scheme	-1= Contains error, 1= No error in header field.
25	"Content_Language" header field status	Contains information about the "Content_Language" header fields. This feature is used to detect any error in this field, such as unauthorized scheme	-1=Contains error, 1=No error in header field
26	Combination of *Request-line* & *Response-Line*	This feature contains information about the presence of both *Request-line* and *Response-line* in the same message	-1=Contains error, 1=No error