HELSINKI UNIVERSITY OF TECHNOLOGY
Faculty of Electronics, Communications and Automation

Ove Liljeqvist

Service availability for the Session Initiation Protocol in a hostile environment

Master's Thesis
Espoo, November 6, 2009

Supervisor Professor Tuomas Aura

HELSINKI UNIVERSITY OF TECHNOLOGY

Abstract of the Master's Thesis

Author: Ove Liljeqvist
Name of the Thesis: Service availability for the Session Initiation Protocol in a hostile environment
Date: 6.11.2009 **Number of pages:** 10 + 67

Faculty: Faculty of Electronics, Communications and Automation
Professorship: T-110 Data Communications Software

Supervisor: Professor Tuomas Aura

The evolution of Voice over IP (VoIP) has brought several advantages for both end-users and service providers, but at the same time new threats have emerged. Denial of Service (DoS) attacks are one of the major threats facing users in the Internet, aiming to disrupt services offered by a network or server.

The Session Initiation Protocol (SIP) is a common choice for establishing, modifying and terminating real-time voice or video sessions. The main objective of this thesis is to evaluate the possibilities of providing real-time SIP-based services in a potentially hostile environment, which the Internet clearly is.

The work is based on a literature study including Internet publications, journal and conference articles. Different approaches to mitigate the effect of DoS attacks are assessed from the viewpoint of a service provider. The available methods and solutions are compared according to their feasibility of implementation and effectiveness against attacks in the context of SIP-based services.

The key findings are that although the DoS attacks are often difficult to protect against, an effective defense strategy includes using a combination of different defense methods, i.e., a layered defense model. Also, the selected defense methods should be scaled based on the perceived risks and adapted to the prevailing conditions under which the service provider operates.

Keywords: Session Initiation Protocol (SIP), Denial of Service (DoS), VoIP, availability
Language: English

TEKNILLINEN KORKEAKOULU

Diplomityön tiivistelmä

Tekijä: Ove Liljeqvist
Työn nimi: SIP-palveluiden saatavuus vihamielisessä ympäristössä
Päivämäärä: 6.11.2009 **Sivumäärä:** 10 + 67

Tiedekunta: Elektroniikan, tietoliikenteen ja automaation tiedekunta
Professuuri: T-110 Tietoliikenneohjelmistot

Työn valvoja: Professori Tuomas Aura

IP-puheen kehitys on suonut monia etuja sekä loppukäyttäjille että palveluntarjoajille, mutta samaan aikaan uusia uhkia on ilmaantunut. Palvelunestohyökkäykset ovat iso uhka käyttäjille Internetissä. Näillä hyökkäyksillä pyritään häiritsemään verkon tai palvelimen käyttäjille tarjoamia palveluja.

SIP-protokollaa käytetään yleisesti luomaan, muokkaamaan ja päättämään reaaliaikaisia puhe- tai videoyhteyksiä. Tämän diplomityön päätavoitteena on arvioida mahdollisuuksia reaaliaikaisten SIP-pohjaisten palvelujen tarjoamiseen mahdollisesti vihamielisessä ympäristössä, jollainen Internetkin on.

Työ on tehty kirjallisuuskatsauksena, jossa aineistona on käytetty Internet-, aikakausi- ja konferenssijulkaisuja. Erilaisia ratkaisumalleja palvelunestohyökkäysten torjumiseen arvioidaan palveluntarjoajan näkökulmasta. Saatavilla olevia keinoja ja ratkaisuja verrataan keskenään huomioiden niiden toteuttamiskelpoisuus ja tehokkuus hyökkäysten estämisessä SIP-pohjaisten palvelujen yhteydessä.

Vaikkakin palvelunestohyökkäyksiltä on vaikeaa suojautua, keskeinen huomio on, että tehokas puolustusstrategia on käyttää eri suojakeinojen yhdistelmää, eli kerroksellista suojamallia. Valitut torjuntamekanismit tulisi lisäksi mitoittaa arvioitujen riskien ja palveluntarjoajan ympäristön mukaan.

Avainsanat: Session Initiation Protocol (SIP), palvelunesto (DoS), VoIP, saatavuus
Kieli: englanti

Preface

It has been a long journey to finally end up writing the preface for this thesis. A long break in the studies did not really help in starting this project, but the writing process got easier as the thesis advanced.

I would like to thank my supervisor, Professor Tuomas Aura, for his indispensable guidance and comments. I also appreciate the proofreading help I got from William Martin and my sister. In addition, I wish to express my gratitude to my parents for not letting me forget this thesis and my whole family for their continued love and support.

Last but not least, I want to thank my dear grandmother for always being there for me and having strong faith in me.

Espoo
November 6, 2009

Ove Liljeqvist

Table of Contents

List of Figures

List of Tables

Abbreviations

3GPP	Third Generation Partnership Project
AoR	Address-of-Record
AS	Autonomous System
B2BUA	Back-to-Back User Agent
BGP	Border Gateway Protocol
CA	Certification Authority
CRLF	Carriage Return Line Feed
DDoS	Distributed Denial of Service
DNS	Domain Name System
DRDoS	Distributed Reflector Denial of Service
DTLS	Datagram Transport Layer Security
FEC	Forward Error Correction
FQDN	Fully Qualified Domain Name
HTTP	Hyper Text Transfer Protocol
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection System
IETF	Internet Engineering Task Force
IM	Instant Messaging
IP	Internet Protocol
IPS	Intrusion Prevention System
IPSec	Internet Protocol Security
IPX	Internetwork Packet Exchange
ISO	International Organization for Standardization
ISP	Internet Service Provider
LAN	Local Area Network
LDAP	Lightweight Directory Access Protocol

LFU	Least Frequently Used
MD5	Message-Digest algorithm 5
MIKEY	Multimedia Internet KEYing
MIME	Multipurpose Internet Mail Extensions
MSC	Message Sequence Chart
MTBF	Mean Time Between Failures
MTTR	Mean Time To Restore
NAPTR	Naming Authority Pointer
NAT	Network Address Translation
OS	Operating System
P2PSIP	Peer-to-Peer Session Initiation Protocol
PKI	Public Key Infrastructure
PPS	Packets Per Second
PSTN	Public Switched Telephone Network
QoS	Quality of Service
RFC	Request For Comments
RON	Resilient Overlay Network
RPF	Router-based Packet Filtering
RSVP	Resource Reservation Protocol
RTCP	Real-Time Control Protocol
RTP	Real-Time Transport Protocol
RTSP	Real-Time Streaming Protocol
SAVE	Source Address Validity Enforcement Protocol
SDP	Session Description Protocol
SIP	Session Initiation Protocol
S/MIME	Secure / Multipurpose Internet Mail Extensions
SMTP	Simple Mail Transfer Protocol
SQL	Structured Query Language
SRTP	Secure Real-Time Transport Protocol

SRV	Service
TCP	Transmission Control Protocol
TTL	Time-to-Live
TLS	Transport Layer Security
UA	User Agent
UAC	User Agent Client
UAS	User Agent Server
UDP	User Datagram Protocol
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
UTF	Unicode Transformation Format
VoIP	Voice over IP
X.25	CCITT Packet Switching Protocol

1 Introduction

1.1 Background

Since Alexander Graham Bell invented the telephone in 1876, circuit-switched networks (also called as Public Switched Telephone Network, PSTN) have been dominating the world of telecommunications. Nowadays, voice is increasingly being transmitted in packet-switched networks. The term "packet-switched" refers to the type of network in which relatively small units of data (packets) are routed through a network based on the destination address contained within each packet. Breaking communication down into packets allows the same data path to be shared among many users in the network. The same is not possible in the circuit-switched network in which a physical path is obtained for and dedicated to a single connection between two end-points in the network for the duration of the connection. During that time, no one else can use the physical lines involved.

Internet, being a packet-switched network, uses a protocol called IP (Internet Protocol) for routing data packets to the correct destination. To achieve better efficiency (and cost savings) as compared to the traditional PSTN, a standard called the Session Initiation Protocol (SIP) has evolved[1]. Developed by the Internet Engineering Task Force (IETF), SIP enables real-time voice communications over the Internet, i.e., "Voice over IP" (VoIP). Besides voice, SIP supports multimedia (e.g. video and data) conferencing and a myriad of other use cases, including instant messaging and online gaming.

1.2 Problem Statement

There is a clear tendency that communications are moving away from closed networks towards open networks. Consequently, malicious activities like spreading viruses or stealing confidential information are on the rise. In a closed PSTN environment, these harmful acts are harder to conduct than in an open Internet environment. Openness gives flexibility, but at the same time enables any end point to communicate freely with (and possibly attack) almost any other end point. This problem intensifies with the fact that it is relatively easy to hide or forge one's true identity over the Internet.

[1] The first SIP standard (RFC 2543) was published in 1999 (Schulzrinne, 2008).

As an application operating on the Internet, SIP is not immune to hostile acts. In addition to the protocol (application) specific attacks, SIP suffers from the common vulnerabilities associated with any Internet service. Attacks can be targeted at the underlying network and transport protocols or on the operating system of the application. Similarly, gateways and servers that the application depends on are at risk. (Sicker and Lookabaugh, 2004)

The Denial of Service (DoS) attacks are of a special concern, since a real-time data stream, such as VoIP, is more easily disrupted than regular Internet applications (e.g. www and email). Even a small disturbance in the traffic can be noticeable for the session participants. Besides, it is not simple to secure a protocol like SIP. Typically, users are communicating directly or using intermediaries with no previous trust relationship (RFC 3261).

Although the provision of secure Internet services is much more complicated for a SIP environment, security and Quality of Service (QoS) requirements are often expected to be equivalent to those in PSTN networks. If SIP is going to replace PSTN as the carrier for voice, security characteristics that should be guaranteed include high service availability, fault tolerant operation, and protection for the traffic flowing between users and between a user and a network (Salsano et al., 2002).

1.3 Objectives and Methodology

The main objective of this thesis is to evaluate the possibilities of providing real-time SIP-based services in a potentially hostile environment, which the Internet clearly is. Although SIP is not limited to voice transmission, in the context of this thesis voice (i.e. VoIP) is a good example of real-time traffic and allows comparison to the PSTN environment.

The work is based on a literature study consisting of IETF Request For Comments (RFCs), Internet publications, journal and conference articles. We explore the available protection mechanisms against DoS attacks and compare them according to their feasibility of implementation (cost, effort etc.) and effectiveness in the context of SIP-based services.

This thesis concentrates on DoS because it is the author's opinion that a DoS attack is significantly easier to conduct (and therefore a more potential threat) than, for instance, eavesdropping or unauthorized tampering of the SIP signaling or the media stream. In general, DoS attacks can be launched from anywhere, whereas eavesdropping on a specific call would require the attacker to be able to position himself on a route between the participants of a call[2], which is much more difficult. To cover the matter fully, DoS attacks requiring the latter ability (attacker between participants) are also described in this thesis.

1.4 Structure

This thesis is divided into six chapters. The remaining chapters are briefly summarized below.

Chapter 2 defines the concept of service availability and different types of DoS attacks are discussed. Also, prevalence of the attacks is estimated and motivations for these attacks are considered.

Chapter 3 introduces the SIP protocol and describes network entities, messages and protocol operation. In addition, other protocols typically used in conjunction with SIP are briefly discussed. Finally, some security mechanisms available for SIP are introduced.

Chapter 4 focuses on the DoS attacks that can be mounted against SIP. A malicious user can exploit the same common vulnerabilities associated with any Internet service, but SIP has its own vulnerabilities that can be taken advantage of.

Chapter 5 presents different methods for protecting against the DoS attacks. Both application independent and SIP-specific protection possibilities are investigated. Afterwards, the countermeasures maximizing the protection against the attacks while keeping the costs low are recommended.

Chapter 6 concludes the thesis and provides a summary of the most essential findings. Further, suggestions for future work are made.

[2] Known as a man-in-the-middle attack.

2 Service Availability and DoS Attacks

This chapter describes the meaning of service availability and presents different ways of ensuring availability in a normal case, i.e., in the absence of deliberate attacks. Afterwards, Denial of Service (DoS) variations are discussed and examples of DoS attacks are presented. Countermeasures against DoS attacks are proposed in Chapter 5.

2.1 Definitions

Dahlin et al. (2003) define the service availability and unavailability in the following way: "*A service is available to a client when that client can communicate with it*" and "*A service is unavailable to a client when that client cannot communicate with it due to a network or end-host failure*". Similarly, Jiang and Schulzrinne (2003) report that availability can also be defined using the terms Mean Time Between Failures (MTBF) and Mean Time To Restore (MTTR):

$$Availability = \frac{MTBF}{MTBF + MTTR} \tag{1}$$

On the other hand, in the telephony or VoIP context, availability is measured as a probability of a call being established on the first attempt:

$$Availability = \frac{Number\ of\ successful\ calls}{Number\ of\ first\ call\ attempts} \tag{2}$$

Equations are clearly different at first glance, but over the long run they yield effectively the same results.

Commercial service providers often advertise 99,999 % ("five-9's"[3]) server availability, but that is not the same as 99,999 % service availability. Server availability figures do not take into account types of failures that can occur within the end-to-end route from the user of the service to where that service is hosted (actual server). (Dahlin et al., 2003)

Telecommunication equipment vendors also advertise the five 9's availability. However, this figure is really for the local switching equipment, not across the entire PSTN. The rest of this section discusses the end-to-end availability model developed by Jiang and

[3] 99,999 % availability means there is five minutes of unavailability per year.

Schulzrinne (2003) and adapted for SIP by the author. The end-to-end service availability in the Internet for the SIP service can be defined as follows:

$$A_{e2e} = A_{UA1} * A_{local1} * A_{network} * A_{local2} * A_{UA2} \quad (3)$$

where A_{UA1} and A_{UA2} are availability figures for the end points. These end points being typically software-based, their availability would depend on the stability of the software running the SIP service. Similarly, A_{local1} and A_{local2} correspond to the availability figures of the end points' local proxy servers used for routing the calls between the caller and callee. Proxy server functionality is described in more detail in Section 3.2.2. It should be noted that in some circumstances calls (sessions) can be set up directly between participants without the use of proxies. Finally, $A_{network}$ is the availability figure for the network (or networks) between the end points. Given that the end points are geographically separated (e.g. not within the same city), it is likely that the availability components are independent. In certain cases (e.g. during a global worm outbreak), the same external factor can have a negative impact on more than one availability component of the Eq. (3) at the same time.

A factor not included in end-to-end availability figure defined by Eq. (3), is end-user experience. Users are generally more troubled with longer outages in a call when the total duration of disruptions remains the same. Still, a call with low quality is better than not being able to establish a call at all. To combat the most common cause for low quality, namely packet loss, one can use techniques such as Forward Error Correction (FEC). In practice, if outages last longer than a few seconds, FEC does not bring any improvement. Packet loss and delay are to be expected, as Internet provides only best effort delivery.

According to research by Jiang and Schulzrinne (2003), the overall service availability for VoIP in the Internet was estimated as 99,53 %. This was the call success probability as defined by Eq. (2). Measurement nodes were scattered across the US, Europe and Asia. Long outages during a call cause users to abort the call, i.e., hang up the phone. If this call abortion probability is taken into account, the same study estimated the net service availability to be 98 %, which is notably lower than the service offered by PSTN (availability measured to be between 99,9 % and 99,99 %). In contrast, the availability of the wireless PSTN (i.e. mobile telephony) ranges from 97,1 % to 98,8 % according to a survey made in United Kingdom (Oftel, 2002). Although these results represent only a

single study, this estimate (98 %) gives a rough idea for SIP service availability in the Internet before DoS attacks are discussed.

2.2 Improving Service Availability

In addition to FEC, there are other means of improving the end-to-end service availability. On the server level (i.e. $A_{local1,2}$ components in Eq. (3)), availability can be increased with clustering. Requests can be distributed to all active cluster members (*load balancing cluster*) or to only one cluster member, which is active at a time (*hot standby cluster*). Essentially, members of a cluster act as one logical entity towards the end-users and if one member fails, other members take over and handle the subsequent communication. In SIP, Service (SRV) records of Domain Name System (DNS) can be used for providing high availability and load balancing for SIP servers (see also Section 3.6.2).

On the network level, caching is a commonly used technique for improving the availability of HTTP traffic. Caching is an example of a client-independence technique where local resources are used for providing a (possibly deteriorated) version of a service in case the remote server is unavailable (Dahlin et al., 2003). In a SIP context though, only a fraction of traffic is cacheable. Results of DNS queries made by end points and proxies can be cached (see Section 3.6.2), but the effect on availability is negligible unless parties that are communicated with stay mostly the same. Obviously, real-time data (e.g. voice) cannot be cached.

Since routing problems are estimated to prevent communications between hosts as much as 3,3 % of the time (Dahlin et al., 2003), it is useful to consider techniques which employ alternate network paths for routing around failures ($A_{network}$ component in Eq. (3)). These rerouting techniques include dynamic routing and overlay networks. In a Resilient Overlay Network (RON) architecture proposed by Andersen et al. (2001), an application layer overlay is placed on top of the existing Internet routing system. RON enables the detection and recovery of path outages for distributed applications in mere seconds, whereas dynamic routing protocols such as BGP (Border Gateway Protocol) take several minutes to recover.

2.3 DoS Attacks

Internet is an open environment originally designed with scalability in mind. Yet, this openness makes it vulnerable to hostile attacks against infrastructure, i.e., clients, servers, routers or links. Denial of Service (DoS) attacks are one of the major threats facing users in the Internet, aiming to disrupt services offered by a network or server. As the IP protocol, which delivers the traffic in the Internet (see Section 3.4.1), has no globally deployed support for source address authentication, it is trivial for an attacker to fake the source address of packets (also called as *IP spoofing*). Thus, it can be difficult to trace the attack traffic to the real source. Moreover, attack traffic can mimic legitimate requests so closely that it is difficult to distinguish the malicious requests from the valid ones. (Handley and Greenhalgh, 2004), (Peng et al., 2007)

2.3.1 Classes of Attacks

Moore et al. (2006) categorize DoS attacks into two classes. They are either *logic* or *resource* attacks. Some attacks are a combination of logic and resource attacks. Logic attacks attempt to exploit software vulnerabilities trying to crash the remote system or inflict degraded performance. One famous example of such an attack is the "Ping-of-death", where large Internet Control Message Protocol (ICMP) packets are sent to the victim. Due to their size, these packets are fragmented into multiple datagrams and when the target host reassembles them, it might crash or reboot. Ping-of-death attacks have only historical relevance, since the underlying vulnerability has been widely patched.

Protocol attacks can be considered as a one form of logic attack. These types of attacks try to exploit a specific protocol feature or a bug in the implementation.

Resource attacks described in (Peng et al., 2007) try to deplete target system resources, for example, CPU capacity, memory, bandwidth (i.e. Internet link capacity) or disk space. Massive volumes of superfluous traffic are sent as rapidly as possible towards the victim overwhelming its ability to serve legitimate users. A classic example is the "SYN flood", where a continuous stream of Transmission Control Protocol (TCP) SYN packets are sent to a TCP port on which the target host is listening. For every TCP SYN request received, the victim is forced to search for a match in the existing connections. If there is no matching connection, a response (TCP SYN/ACK) is sent back and a new data structure is allocated until the victim's memory stack is full. Traditionally, a SYN flood attack has used IP spoofing to conceal its source and to ensure responses from the

victim are unanswered, but the attack can be initiated from compromised hosts using real source addresses provided that these hosts are configured to ignore responses from the target.

"Smurf" (CERT, 1998) is a combination of a protocol (logic) and a resource attack. In this attack, ICMP echo request (i.e. ping) packets with spoofed source addresses of the target are sent remotely to an IP broadcast address of a particular network. Once a packet is sent to an IP broadcast address, it is delivered to all hosts on that network, if routers are configured to relay IP-directed broadcast traffic. These intermediary hosts then send ICMP echo replies overwhelming the victim. Nowadays, attacks based on smurf are rare as hosts seldom respond to pings sent to broadcast addresses and routers do not commonly forward packets to broadcast addresses.

As suggested by Peng et al. (2007), attack power can be defined as the level of resources consumed at the target by the attack. In fact, attack power consists of two factors. Traffic volume (or packet rate) depicts the number of packets per second (pps) received by the victim. The second factor is the amount of resources a single packet uses at the target system.

While known vulnerabilities can be patched to mitigate the threat posed by the logic attacks, defending against the resource attacks is very difficult. Various defense mechanisms are discussed in more detail in Chapter 5.

2.3.2 Distributed Denial of Service (DDoS)

Whenever a DoS attack is dispersed among multiple sources, it is called a Distributed Denial of Service (DDoS) attack. The combined attack volume can be as high as 10 Gb/s (Peng et al., 2007). Consequently, DDoS is arguably the most powerful type of a DoS attack. Typically, an attacker harvests a large group of hosts by compromising vulnerable systems and installs attack tools in these systems (Moore et al., 2006). To compromise the systems, the attacker can use viruses, worms and automated scanning from already hijacked end points (Handley and Greenhalgh, 2004).

These "zombies" or "bots" can then be instructed to attack a specific target (see Figure 1). Additionally, the attack can use spoofed source addresses to make it harder for the victim to trace the source of the traffic. However, as the size of a bot army ("botnet")

can exceed 300,000 hosts and can be geographically distributed (Peng et al., 2007), knowing the real IP addresses provides only a marginal benefit for the victim.

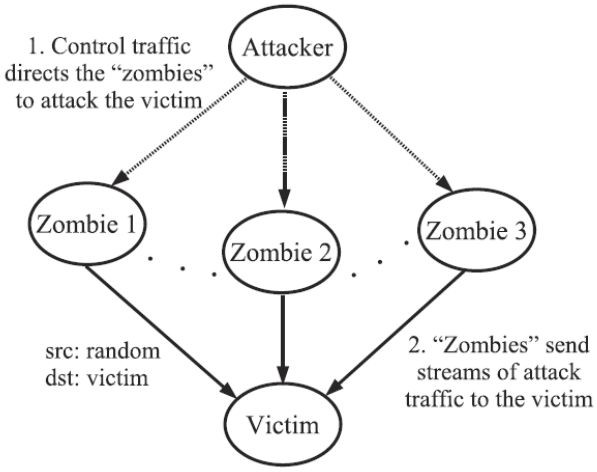

Figure 1: DDoS Attack Structure (Source: Peng et al., 2007)

A special case of a DDoS attack is Distributed Reflector Denial of Service (DRDoS), which further obscures attack sources by relaying attack traffic through third parties (called as *reflectors*). These reflectors can be, for example, routers or DNS servers, which receive attack traffic with spoofed source address of the victim. Since DRDoS attacks can amplify[4] the malicious traffic towards the victim and are extremely hard to trace, they pose a very serious threat to the Internet hosts. (Peng et al., 2007)

2.3.3 Contributing Factors

As discussed earlier, the Internet only provides best effort service, meaning that all users share the same resources. In other words, one user has the ability to affect the quality of service for other users. Resource attacks are based on this Internet design principle.

To simplify the core networks, intelligence in the Internet is pushed to the network edges (i.e. communicating hosts). Routers only deliver packets according to the destination IP address without needing to understand applications above the network layer. For this reason, there is no hop-by-hop authentication and all packets are forwarded to the destination address even if they are then discarded by the receiving host. Further, the

[4] In the DNS amplification attack, small DNS request packets generate significantly larger response packets. In fact, a theoretical amplification factor can be as high as 73 (Vaughn and Evron, 2006).

path between the source and destination can vary. As a result, the Internet is resistant to certain network outages, but at the same time tracing of packets is very difficult. (Peng et al., 2007)

DoS attacks commonly cross different administrative and jurisdictional borders. This makes it even harder to monitor or trace the attacks, as there is no central management in the Internet. In other words, no single organization has a complete view to a typical DoS activity. (Lipson, 2002)

Finally, as hosts are highly dependent on core infrastructure services (e.g. routing, DNS name resolving), a successful attack does not need to target its victim directly (Peng et al., 2007). When an attack on the infrastructure succeeds, there are most likely multiple victims. The attack against all DNS root servers in 2002 was a good reminder of the criticality of infrastructure services.

2.4 DoS Activity in the Internet

To get an idea of the prevalence of DoS attacks in the Internet, Moore et al. (2006) made a quantitative analysis of DoS activity during a three year period, from 2001 through 2004. Using a technique called “backscatter analysis”, they estimated that there were 68,700 attacks against 34,700 distinct hosts. The majority of these victims were broadband users and small companies.

Moore et al. (2006) also suggest that many attacks are scripted, since there were evidence of periodic patterns in the attacks. In addition, 93 % of attacks were TCP-based. Most attacks were relatively short. In fact, 80 % of attacks lasted less than half an hour.

Motivations for DoS attacks range from mischief to ethnic or political disputes. Also, some of the attacks are clearly motivated by a financial benefit (Poulsen, 2004).

2.5 Summary

This chapter presented the definition for end-to-end service availability and discussed some means of improving it. According to one study, a maximum of 98 % end-to-end availability can be expected for the VoIP service in the Internet. This also gives a rough idea for the SIP service availability when a real-time traffic is considered.

Next, DoS attack classes were introduced and some examples were given. IP spoofing was found to hinder the tracing of attacks to their real sources. DDoS was presented as the most serious threat. Finally, DoS attack prevalence was estimated and motivations for these attacks were considered.

3 Session Initiation Protocol

This chapter gives an introduction to the Session Initiation Protocol (SIP). The main characteristics such as network elements, message structure and operation of the protocol are described. Security mechanisms available for SIP are also explored. The chapter is mainly based on the IETF RFC 3261 standard.

3.1 Overview

SIP is an open standard developed by the IETF. It is an application-layer control protocol that can establish, modify and terminate multimedia sessions. A multimedia session can be an audio (i.e. VoIP) or video call, Instant Messaging (IM) or a similar application. SIP supports both unicast and multicast sessions.

SIP helps parties wishing to communicate to locate each other on the Internet. In addition, it enables those parties to negotiate how they are going to communicate (e.g. which codecs[5] to use) by exchanging supported media parameters. After a session is set up, it is possible to add media and participants to an ongoing conversation. Although SIP is not just a simple telephony application protocol, typically it is used to establish and tear down real-time voice or video sessions. It is worth noticing that apart from being used in existing 3G cellular networks, next generation (4G) networks will use SIP as a primary signaling protocol (3GPP).

In essence, SIP is a framework for the development of communications applications. Despite the fact that a considerable effort has gone into specifying and implementing SIP using a service-provider model, this framework allows alternative models. Consequently, there is currently an active IETF working group defining the Peer-to-Peer Session Initiation Protocol (P2PSIP) which does not depend on centralized services. (P2PSIP), (Sparks, 2007)

Using 8-bit Unicode Transformation Format (UTF-8) encoding in its syntax, SIP is modeled upon other Internet protocols such as the Hyper Text Transfer Protocol (HTTP) and the Simple Mail Transfer Protocol (SMTP). Therefore, SIP is a text-based stateless protocol. This makes it simple and flexible and allows easy implementation in

[5] Voice or video encoding algorithms

languages such as Java or Perl. It is also possible to add new capabilities via extensions to the protocol.

3.2 Network Entities

The main logical components within a conventional[6] SIP architecture are a User Agent (UA) and a SIP server. As SIP defines only logical entities, there does not need to be any physical difference between the server entities and different logical servers can reside in the same physical system.

3.2.1 User Agent

UAs are the SIP endpoints able to both generate and answer requests. A User Agent itself has a client element, a User Agent Client (UAC) and a server element, a User Agent Server (UAS). IP-phone is a typical UA.

UAC is a client application that initiates a SIP request, whereas UAS is a server application that contacts a user when a SIP request is received and returns a response on behalf of the user.

UAs are the only entities within a SIP architecture where signaling requests and media streams converge. A peer-to-peer relationship where messages are exchanged between two UAs is referred to as a SIP dialog.

3.2.2 Proxy

A proxy server is an intermediate SIP entity acting as a message router. If a proxy knows the exact destination address, it establishes a direct connection to send the message. Otherwise, message is forwarded to another proxy closer to the destination. A proxy can also forward the request to multiple servers at once (a so called parallel forking proxy), in the hopes of contacting the user at one of the locations. Additionally, a proxy can forward the request to one server at a time (a so called serial forking proxy), letting the first request time out before trying the second. (Sparks 2007)

[6] SIP has been modified to suit 3G networks, where it is called as “3GPP SIP” (3GPP). 3GPP SIP is out of the scope of this thesis.

Proxies can be running in either stateful or stateless mode. If stateful, a proxy remembers transaction state for each incoming request and corresponding outgoing requests sent as a result of processing the incoming request. A stateful proxy may also retain a state for a SIP dialog. On the other hand, a stateless proxy forgets all the information it has received once an outgoing request has been generated. A forking proxy cannot be stateless because it needs to perform a filtering operation, returning (in general) one response out of the many it receives (Schulzrinne, 2008).

A request from A to B can be routed through several proxies. It is desirable to force the response(s) to such a request to follow the same route back. For instance, a proxy might be billing the call, or controlling a firewall, and needs to have access to all the information regarding the call. Consequently, these kinds of proxies must be stateful. A Via header (see Section 3.5.3 for more details) traces the path of the request, allowing the responses to find their way back. In fact, this also helps to avoid routing loops (each proxy checks whether it is already in the Via list). Each time a proxy forwards a request, it must add itself to the beginning of the list of forwarding proxies recorded in the Via headers. When a proxy forwards a reply, it reverses the process and removes its name from the list. (Hersent et al., 2000)

3.2.3 Registrar

A registrar is a server that accepts REGISTER requests for a particular domain. It is typically co-located with a proxy serving the same domain.

A registrar maintains a mapping between an Address-of-Record (AoR) and a zero or more contact addresses. These entries (bindings) are updated any time when a user joins or leaves a SIP network. In other words, a registrar keeps track of the current location of a user providing a basic mobility service. For storing this location information, registrars use databases known as location servers.

3.2.4 Location Server

A location server is a network entity providing user location information (AoR entries) to a redirect server or proxy server. Since the SIP standard does not specify how to interact with a location server, some other protocol (e.g. Lightweight Directory Access Protocol, LDAP) has to be used in the communication with it.

3.2.5 Redirect Server

Redirect servers help in locating users by providing alternative set of addresses for a caller to try. These addresses are contained in a 3xx class response (see Section 3.5.2 for more details about different response classes).

Unlike a proxy server, a redirect server does not initiate any SIP requests of its own. This characteristic makes a redirect server a useful tool in improving the scalability of a SIP network, since it only has to send back a response with the correct location, instead of participating in the whole transaction. (Hersent et al., 2000)

3.2.6 Back-to-Back User Agent

A Back-to-Back User Agent (B2BUA) is an intermediate entity that terminates and re-establishes the SIP signaling and media. This means that a B2BUA has the ability to modify the messages it retransmits, which is why B2BUAs are often used in conjunction with firewall or Network Address Translation (NAT) devices. When sitting on a network boundary, policy control and interoperability concerns are easier to address. (Sparks, 2007)

3.3 Addressing

A Uniform Resource Identifier (URI) is a compact string of characters for identifying an abstract or physical resource. A resource can be anything that has an identity, for instance, an electronic document, an image or a service. Also, a resource can be something not retrievable from the network (persons, corporations). (RFC 2396)

SIP addresses users by an email-like address, called as a SIP URI. The SIP URI has the (simplified) form *sip:user@host* where the user part is a user name or a telephone number. The host part is either a domain name or a numeric network address. When a SIP URI represents a user's logical identity, it is called an Address-of-Record (AoR) and can be thought of as the “public address” (e.g. alice@example.com) of the user. With the help of a location server, a user's current address (e.g. alice@pc22.example.com) can be resolved using AoR-to-contact-URI binding.

SIP also provides a secure URI, called a SIPS URI (e.g. sips:alice@example.com) which allows a secure signaling session to be established using Transport Layer Security (TLS)

between UAs. It should be noted that the current standard does not guarantee that TLS is used end-to-end.

3.4 Other Protocols Involved

SIP is rather independent of the environment because it makes minimal assumptions about the underlying transport and network-layer protocols. In fact, any datagram or stream protocol that delivers a whole SIP request or response can be used. The overall protocol architecture is illustrated in Figure 2.[7]

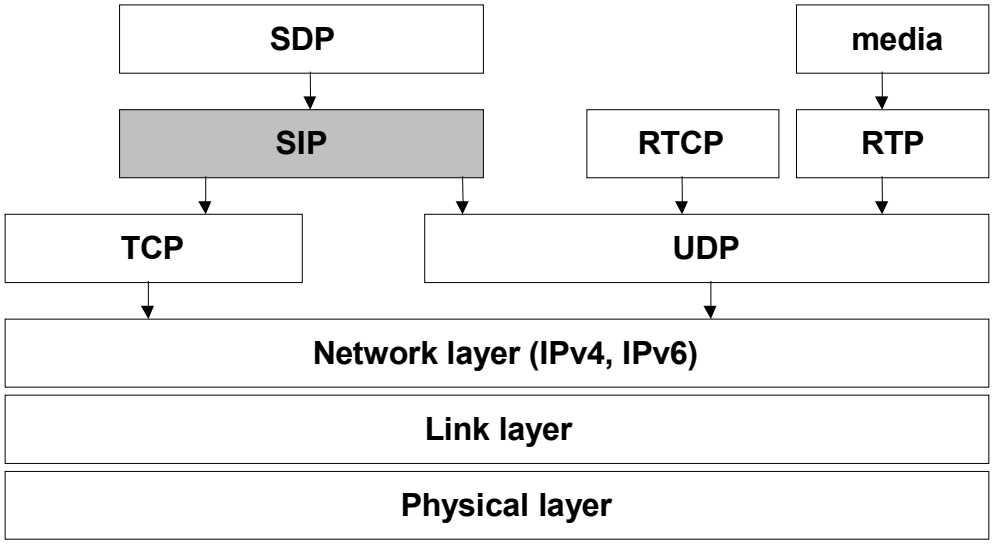

Figure 2: SIP Protocol Architecture

SIP is typically used over the User Datagram Protocol (UDP) or TCP, but it could be run over Internetwork Packet Exchange (IPX), carrier pigeons, frame relay or X.25, to name a few (Schulzrinne, 2008). UDP allows the application to more carefully control the timing of messages and their retransmission, to perform parallel searches without requiring a TCP connection state for each outstanding request, and to use multicast. TCP is needed to be used when UA is handling large messages.

Multimedia architectures using SIP typically incorporate other protocols such as the Real-Time Transport Protocol (RTP) for transporting real-time data, the Real-Time Control Protocol (RTCP) for providing QoS feedback, the Real-Time Streaming Protocol (RTSP) for controlling delivery of streaming media and the Session Description

[7] Secure versions of the protocols shown are discussed in Section 3.7.

Protocol (SDP) for describing multimedia sessions. However, the functionality and operation of SIP does not depend on any of these protocols mentioned.

The most common protocols surrounding SIP are described in a more detail in the next sections.

3.4.1 IP

Although SIP is independent of the underlying network-layer protocol, in practice the Internet Protocol (IP) is the protocol that carries SIP messages from a source to a destination (as well as it transmits most of the other traffic in the Internet).

IP is a packet-based protocol used to exchange data over computer networks. IP handles addressing, fragmentation, reassembly, and protocol demultiplexing. It is the foundation on which all other IP protocols (collectively referred to as the IP protocol suite) are built. As a network-layer protocol, IP contains addressing and control information that allows data packets to be routed. (Stevens, 1994)

IP is a connectionless protocol, which means that there is no established connection between the end points that are communicating. Each packet that travels through the Internet is treated as an independent unit of data without any relation to any other unit of data. (Stevens, 1994)

The most widely used version of IP today is Internet Protocol Version 4 (IPv4). However, IP Version 6 (IPv6) is also beginning to be supported.

3.4.2 SDP

The Session Description Protocol (SDP) defined in (RFC 4566) allows the session participants to negotiate the session details, such as used protocols, codecs, network addresses and ports. Thus, SIP can use SDP to describe the capabilities and media types supported by the endpoints. This description is contained in the SIP message body, which is a Multipurpose Internet Mail Extensions (MIME) data block specified in (RFC 2045). As mentioned earlier, SIP does not carry the actual media content, but a real-time transport protocol (e.g. RTP) is used.

Like SIP, SDP is a text-based protocol using UTF-8 encoding in its syntax. An SDP session description contains:

- Session name and purpose
- Time(s) the session is active
- The media including the session
- Information needed to receive the media (e.g. addresses, ports and formats)

Additional information may include bandwidth information to be used by the session and contact information for the person responsible for the session.

The syntax of the session description consists of a number of text lines in the form <*type*>=<*value*>, where <*type*> is always exactly one character and is case-significant. <*value*> is a structured text string whose format depends on <*type*>. It also will be case-significant unless a specific field states otherwise. Furthermore, session description is divided into session-level parts (details that apply to the whole session and all media streams) and optionally several media-level sections (details that apply to a single media stream). Common type fields are shown in the table below (optional items are marked with a '*').

Table 1: Common SDP Type Fields

TYPE	DESCRIPTION
v	Protocol version
o	Owner and session identifier
s	Session name
c*	Connection information - mandatory if not present in the media-level
t	Time the session is active
m	Media name and transport address
a*	Zero or more media attribute lines

In order to clarify the use of SDP types in actual messages, an example is shown in Figure 3.

```
v=0
o=oliljeqv 2808844564 2808844564 IN IP4 host.example.com
s=Multimedia call example
c=IN IP4 host.example.com
t=0 0
m=audio 49174 RTP/AVP 0
a=rtpmap:0 PCMU/8000
m=video 49170 RTP/AVP 32
a=rtpmap:32 MPV/90000
```

Figure 3: SDP Description Example

3.4.3 TCP and UDP

Both the Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) belong to the IP protocol suite, which (as mentioned earlier) lays a foundation for upper layer protocols. Because SIP does not require any reliable transport protocol, UAs can use UDP instead of TCP. Nevertheless, both TCP and UDP must be supported by the SIP end points.

TCP is a connection-oriented protocol (i.e. reliable protocol). A three-way handshake is used for establishing a connection. For ensuring a reliable delivery of packets between endpoints, TCP uses a technique called positive acknowledgement with retransmission, meaning that acknowledgements are required for all data sent. If the sending host does not receive these acknowledgements from the other end, data is retransmitted. TCP is also responsible for ensuring that a message is divided into the packets that IP manages and for reassembling the packets back into the complete message at the other end. (Stevens, 1994)

UDP is an unreliable connectionless protocol. Unlike TCP, UDP does not provide the service of dividing a message into packets (datagrams) and reassembling it at the other end. Specifically, UDP does not provide sequencing of the packets that the data arrives in. This means that the application that uses UDP must be able to make sure that the entire message has arrived and is in the right order. (Stevens, 1994)

3.4.4 RTP and RTCP

The Real-Time Transport Protocol (RTP) described in (RFC 3550) provides end-to-end delivery services suitable for real-time data, such as audio or video. Services offered include payload type identification, sequence numbering, timestamping and delivery monitoring. RTP is typically used to transport data on top of UDP. It is worth noticing

that no end-to-end protocol, including RTP, can ensure in-time delivery. This always requires the support of lower layers that actually have control over resources in switches and routers (e.g. Resource Reservation Protocol, RSVP).

The Real-Time Control Protocol (RTCP), which is a counterpart for RTP, is based on the periodic transmission of control packets to all session participants. These control packets use the same delivery mechanism as the data packets. The main function of RTCP is to provide feedback on the quality of the data distribution. In other words, with RTCP all participants know how well the others are receiving real-time data. RTCP is also necessary for synchronizing the playback of audio and video streams.

3.5 SIP Messages

SIP is a text-based protocol and uses the International Organization for Standardization (ISO) 10646 character set in UTF-8 encoding. Lines are terminated with Carriage Return Line Feed (CRLF). A SIP message is either a request from a client (i.e. UAC) to a server (i.e. UAS), or a response from a server to a client. Both requests and responses follow a generic-message format, in which the message 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 (Figure 4 provides an illustration). The start line along with the header fields defines the nature of the session in terms of services, addresses and protocol features. The message-body is independent of the SIP protocol and can contain anything that is described using MIME syntax and semantics.

Except for minor differences in character sets and line termination, much of the message syntax is and header fields are identical to HTTP/1.1 specified in (RFC 2616). As a result, SIP can be easily integrated with web servers.

To make SIP signaling more secure, encryption and authentication can be used. For example, it is possible to encrypt messages to prevent packet sniffers and other eavesdroppers from seeing who is communicating with whom (see Section 3.7 for more details).

Figure 4: SIP Message Example

3.5.1 Requests

A request is characterized by a start line, called the Request-Line. As shown in Figure 4, it consists of a method token (e.g. “INVITE”) followed by a Request-URI and the protocol version. There are six different kinds of requests, called methods, defined in (RFC 3261).

- **INVITE:** The INVITE request indicates that the user or service is being invited to start a dialog.

- **ACK:** The ACK method confirms that the client has received a final response (see Section 3.5.2) to an INVITE request. In fact, ACK is only used in conjunction with INVITE.

- **OPTIONS:** A User Agent sends an OPTIONS request to another UA or proxy server to learn its capabilities. The target answers with a list of header fields, such as 'Allow' and 'Supported' (see Section 3.5.3).

- **BYE:** The BYE request is used for terminating a session and the dialog associated with it. It may be issued by either caller or callee.

- **CANCEL:** The CANCEL request aborts a pending INVITE request, but does not affect a completed request or existing sessions. A request is considered completed if the server has returned a final response.

- **REGISTER:** A SIP endpoint uses the REGISTER method to register its current location with a registrar server.

3.5.2 Responses

After receiving and interpreting a request message, the recipient responds with a SIP response message. The first line of a response message contains the protocol version followed by a numeric Status-Code (a 3-digit integer) and a corresponding textual explanation called the Reason-Phrase.

So far six categories of Status-Codes have been defined, depending on the first digit as shown in Appendix A. This kind of classification makes it easier to add new codes because SIP applications do not need to understand the meaning of all response codes. They have to be able to recognize the class of the response and treat any unrecognized response as being equivalent to the x00 response code of that class.

A request (and its retransmissions) together with the responses triggered by that request make up a SIP transaction. Responses with Status-Codes 2xx, 3xx, 4xx, 5xx or 6xx are called as final responses, because they terminate the SIP transaction. In contrast, the 1xx responses are called as provisional and do not terminate the SIP transaction. Final responses can be further divided into positive final responses (class 2xx) indicating success and negative final responses (classes 4xx to 6xx) indicating an error condition. Redirection responses (class 3xx) imply that the target has moved outside of its usual location.

Provisional responses indicate that the request was received by the callee (or SIP server) and the transaction is in progress (e.g. “100 Trying”). Provisional responses are only sent in response to INVITE requests.

3.5.3 Headers

Headers can be used to specify such parameters as caller, callee, the path of the message, type and length of the message body (see Figure 4 for an example). Each header field consists of a name followed by a colon and the field value (*field-name: field-value*). An application must ignore header fields not defined in (RFC 3261) that it does not understand.

Next, some of the most fundamental headers are explained in more detail. They are mandatory or take place frequently in the messages.

- **Call-ID:** The Call-ID parameter uniquely identifies a particular invitation (and a resulting dialog) or all registrations of a particular client. Call-ID is generated as a combination of a random string and UA's host name or IP address (e.g. a84b4c76e66710@pc33.atlanta.com).

- **Contact:** Provides a SIP or SIPS URI where the user can be reached for further communications. It is usually composed of a user name and a fully qualified domain name (FQDN), e.g. sip:alice@pc33.atlanta.com.

- **Content-Length:** The Content-Length field indicates the size of the message body in octets. If no body is present in a message, then the Content-Length is set to zero.

- **Content-Type:** This field defines the information type of the message body (e.g. application/sdp).

- **CSeq:** Every request has to have a CSeq header field, which contains a sequence number and the method name (e.g. 4711 INVITE). The sequence number is incremented when a new request is generated within a dialog.

- **From:** This field specifies the initiator of the request and must be present in all requests and responses. It contains a SIP or SIPS URI and an optional display name. This header also contains a tag parameter consisting of a random string (e.g. Anonymous <sip:c8oqz84zk7z@privacy.org>;tag=hiy8).

- **Max-Forwards:** Defines a limit for the number of hops via proxies or gateways a request can make on the way to its destination. It consists of an integer that is decremented by one at each hop. Recommended initial value is 70.

- **To:** Indicates the original destination of the request and must be present in all requests and responses. Similar to From-field, this field contains a SIP or SIPS URI and an optional display name (e.g. sip:+12125551212@host.phone2net.com).

- **Via:** The Via field indicates the path taken by the request so far and shows the path that should be followed in routing responses. It also contains a branch parameter that identifies a transaction generated by the request. The branch parameter is used by proxies to detect loops.

3.6 Protocol Operation

This section explains the basic protocol functionality and operation using Message Sequence Charts (MSCs). MSC is a graphical and textual language for the description and specification of signaling between system entities (MSC).

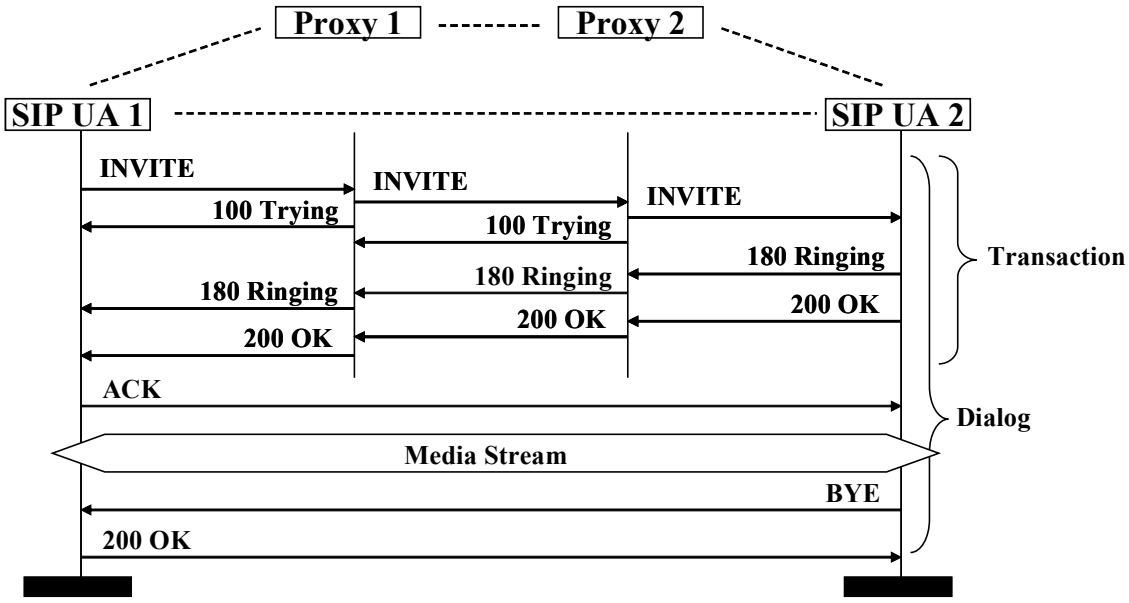

Figure 5: SIP Trapezoid

3.6.1 Session Setup and Teardown

When a user wants to call another user[8], the caller initiates the call with an INVITE request. The request contains enough information for the called party to join the session. In the simplest case, if the client knows the location of the other party it can send the request directly to its IP address. However, in the currently deployed SIP environments, most signaling between endpoints involves one or more SIP proxies (Sparks, 2007). Hence, a typical message flow during a SIP session resembles a trapezoid (as shown in Figure 5) implying that a media stream usually takes a different path than signaling.

[8] SIP is capable of establishing multiparty conferences, but to simplify illustrations, this thesis assumes there are two participants in all sessions.

Before a session can be established, both parties need to be registered (see Section 3.6.4). In Figure 5, when proxies are processing the INVITE request, a “100 Trying” response is sent back to the caller to indicate that the request is being routed to its destination. SIP typically uses port 5060 for signaling between SIP servers and endpoints. Section 3.6.2 provides more details on how UA finds SIP servers (in this case a SIP proxy).

When the INVITE request reaches its final destination, a “180 Ringing” response is sent to the originator of the session. After the callee answers, a “200 OK” response is routed back to the caller, who acknowledges the reception of this final response with ACK. Now, media packets can be exchanged between the two participants of the session. In this case UA 2 ends the conversation (e.g. hangs up the phone) and BYE is sent to the caller. The session ends when a “200 OK” message is sent as a response to this BYE request.

3.6.2 Locating a SIP Server

The SIP server discovery process is specified in (RFC 3263). UA uses DNS procedures to resolve the host part of the SIP URI (i.e. Request-URI) into the IP address, port, and transport protocol of the next hop contact. SIP servers use essentially the same procedures for locating other SIP servers.

If the URI does not specify a transport protocol or port, and the target is a host name, the client performs a Naming Authority Pointer (NAPTR) query for the domain in the URI to determine which transport protocol to use. Otherwise, UA should generally use UDP for a SIP URI and TCP for a SIPS URI. If a NAPTR query is made, its results are used in a SRV query to find out a port and the name of the particular server to use.

It is worth noticing that using SRV records with multiple servers provides a basis for load balancing and high availability for SIP servers.

3.6.3 Locating a User

Once the transport address of the server has been resolved, that server will be the destination of the initial INVITE message. If it is not the final destination of the call, it will redirect the request to the called party. This can be done either by instructing the

caller to send a new INVITE to another location using the 3xx class response (in the case of a redirect server, Figure 6) with Contact headers, or by transparently relaying the request to the appropriate IP address (in the case of a proxy, Figure 7). Provisional responses have been left out of the MSCs for conciseness. In both cases, the called party's (UA 2) contact address is asked from the location server (i.e. registrar).

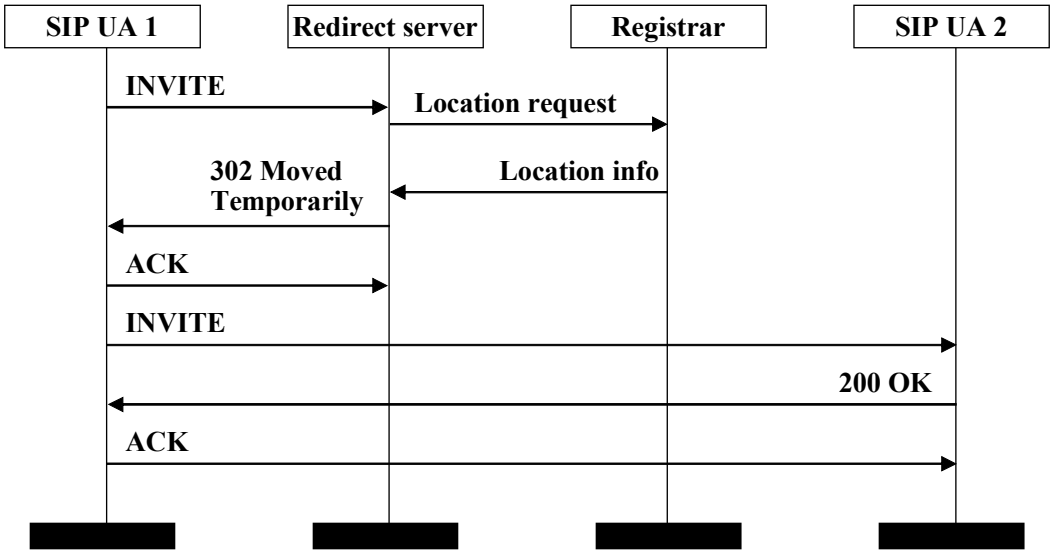

Figure 6: Session Setup Using a Redirect Server

If a UA has associated its AoR with several contact addresses and, therefore, a registrar returns a list of addresses, these addresses can be prioritized using a so called “q” parameter. This parameter indicates a relative preference for a particular address as compared to other AoR-to-contact-URI bindings. INVITE requests can then be sent to these addresses serially until a call is answered or the list is exhausted.

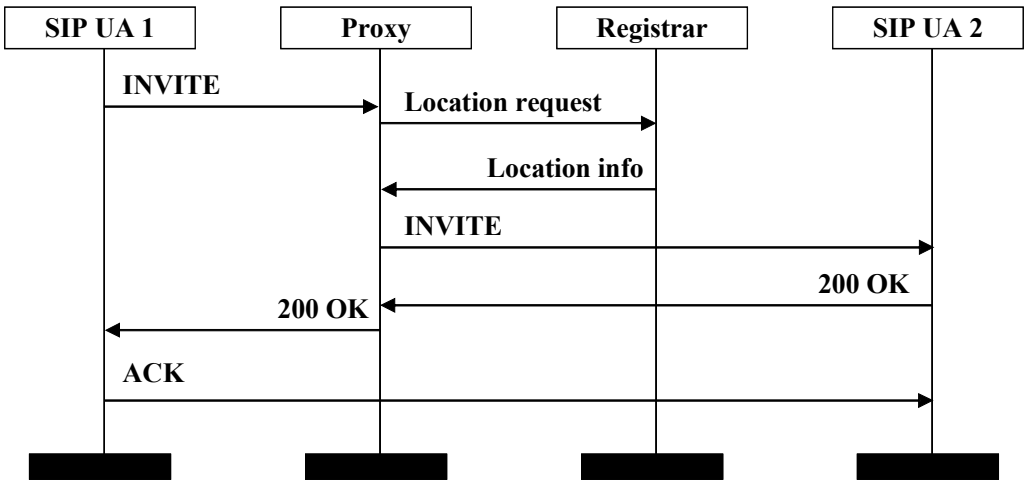

Figure 7: Session Setup Using a Proxy

3.6.4 Registration

User locations can be dynamically registered with a SIP server. In fact, a location server enables UA movement between a number of different end systems over time. A REGISTER request to a registrar allows a client to let a proxy or redirect server know at which address(es) it can be reached. As discussed in the previous section, the location server possibly returns several locations because the user is logged in at several UAs simultaneously or because the location server has inaccurate information.

A registrar may require authentication in order to prevent unauthorized access to a registration database. It searches for the Authorization header field within the REGISTER request, and if not present, responds with a “401 Unauthorized” message. After receiving the 401 response, the UA sends a new request with a suitable authentication information (see Figure 8). More details on authentication and other security mechanisms in SIP can be found from the next section.

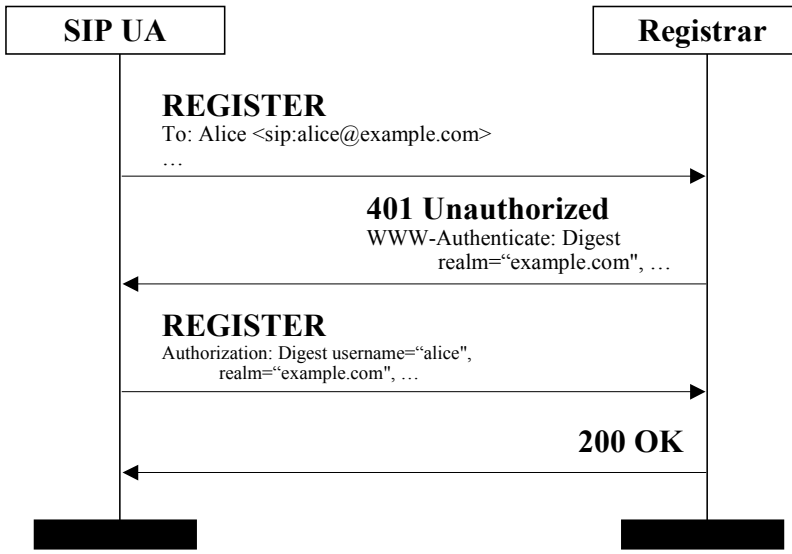

Figure 8: SIP Registration with Digest Authentication

3.7 SIP Security Mechanisms

SIP messages and the related media stream can contain sensitive information about the communication motives and communication content of the individuals. Most of the time this information needs to remain confidential and unchanged. SIP specification (RFC 3261) does not dictate which security schemes should be used. Instead, some well-known mechanisms are suggested. This section describes those and other commonly used mechanisms for ensuring secure communication in a SIP environment.

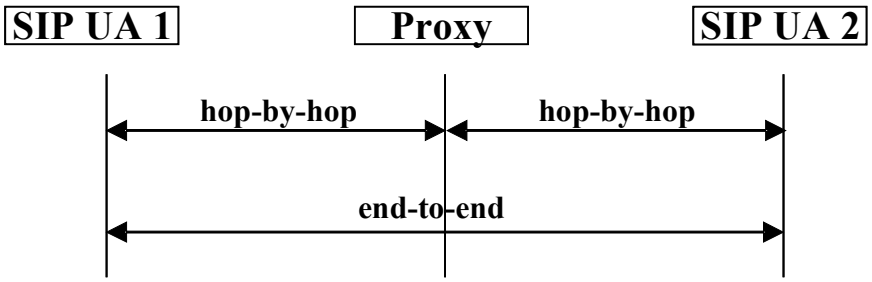

Figure 9: SIP Security Mechanisms

3.7.1 Securing SIP Signaling

According to Sparks (2007), security for SIP signaling messages can be provided either on a hop-by-hop or end-to-end basis as illustrated in Figure 9. For the hop-by-hop protection, TLS or Datagram TLS (DTLS) can be used. TLS usage typically requires that TCP is used as a transport protocol, whereas DTLS supports UDP (RFC 4347).

TLS and DTLS ensure the confidentiality and integrity of messages traversing a particular hop. Further, these transports enable a mutual certificate-based authentication between directly communicating nodes, although UAs rarely have certificates signed by a trusted Certification Authority (CA) needed for strong identity verification.

As the lack of a prevalent Public Key Infrastructure (PKI) is clearly a problem for authenticating end users, SIP supports digest authentication, which closely parallels HTTP digest authentication described in (RFC 2617). With digest authentication a UA can authenticate itself to a SIP server with which it has a pre-existing association (e.g. a proxy in a home network). A limitation with digest authentication is that it provides neither integrity nor confidentiality for SIP messages.

SIPS URI scheme is used to indicate that each proxy along the path to the destination must use TLS. However, the downside is that the end users are forced to trust these intermediate proxies as they have a full access to the contents of the SIP messages routed through them. Besides, SIP standard does not guarantee that TLS is used in every hop.

Another hop-by-hop protection alternative to use with SIP is IP Security (IPSec), which operates in the network-layer. Like TLS, IPSec provides confidentiality and integrity for SIP messages as well as data origin authentication (RFC 2401). The use of IPSec requires an existing trust relationship between communicating nodes or some sort of PKI needs to be in place to support the exchange of cryptographic keys. As opposed to TLS, IPSec does not typically require any integration with a SIP application that it protects.

Parts of the SIP messages not needed for routing[9], as well as MIME bodies, can be encrypted end-to-end using Secure MIME (S/MIME). Also, integrity can be ensured for the SIP message body and most header fields. UAs should possess certificates signed by a trusted CA for a mutual authentication, as well as for secure key exchange.

Given the fact that the From field is easy to forge and UAs commonly lack certificates to prove their identities, another mechanism for secure identification of the originator of a SIP request is needed. This kind of mechanism, namely SIP Identity, is specified in (RFC 4474). This scheme uses a proxy as an authentication service and two new SIP headers,

[9] Request-URI and certain header fields, such as Route and Via, need to be visible to proxies.

Identity and Identity-Info. Since a proxy normally has a certificate, using this certificate it is able to cryptographically assert the identities (AoRs) of end users belonging to its domain when they send requests to users in another domain.

Basically, a hash value of a particular content (including the From header field) of a SIP request is calculated and signed with the proxy's private key for the domain. The signed hash is then inserted into the Identity header. Finally, the proxy inserts the Identity-Info header conveying a reference to its certificate to the request and the message recipient can verify the signature in the Identity header using this certificate.

Hence, there are many security mechanisms for a UA to choose from. Fortunately, a security agreement mechanism described in (RFC 3329) aids in negotiating an appropriate level of security between the UA and its first-hop entity. First, the UA sends a list of of its supported security mechanisms to the server along with its initial request. The server responds with a similar list and the UA then selects the highest-preference security mechanism they have in common. Finally, the UA turns on the selected security and contacts the server again using the negotiated security mechanism.

3.7.2 Securing Media Exchange

For securing the media channel, there are at least two options (Sparks, 2007). The oldest alternative is to use Secure RTP (SRTP) defined in (RFC 3711). SRTP is a framework providing confidentiality, integrity and replay protection for the RTP and RTCP streams. SRTP relies on the secure exchange of keying material but does not dictate which key management solution should be used. One candidate for the key management is Multimedia Internet KEYing (MIKEY) as utilized by Kim et al. (2008).

Besides SRTP, one could use RTP over DTLS (as presented in Tschofenig and Rescorla, 2006) for protecting the media stream. The idea is that normal RTP and RTCP payloads sent inside a UDP packet are also sent inside of a DTLS packet. Similarly as when DTLS is used to secure SIP signaling, end points need trusted certificates in order to validate each other's identities.

3.8 Summary

This chapter gave an introduction to SIP, which is an open, text-based protocol used to establish, modify and terminate multimedia sessions. Definitions were presented in the same way as they appear in the SIP standard (RFC 3261). Also, other protocols typically used in conjunction with SIP were briefly introduced. Especially the presented security mechanisms will serve as a technical background information for the following chapters.

4 DoS Attacks Against SIP

The Session Initiation Protocol (SIP) suffers from the common vulnerabilities associated with any Internet service. The attacks take advantage of these vulnerabilities existing in, for example, a network infrastructure or transport protocols. For example, a DDoS attack could be directed at a SIP entity in the same way that it could target a web server. The author believes that it is the real-time nature of VoIP that makes SIP more vulnerable to Denial of Service (DoS) attacks compared to many other Internet applications. In addition to generic attacks, there are specialized DoS attacks targeting the underlying SIP protocol.

This chapter describes the various DoS attacks against the SIP services. These attacks include flooding which aims to deplete the resources needed by the SIP entities and attacks that either exploit features of the protocol or flaws in the implementation.

Out of all SIP entities possible to attack against, this thesis focuses on SIP proxies as an attack target, since they are generally regarded as the most important assets of a SIP service provider (Sisalem et al., 2006). Further, a proxy is a SIP entity that must be exposed to the Internet in order to accept requests outside the home network. Nevertheless, most attacks presented in this chapter are applicable to UAs as well.

4.1 Resources Targeted

An attacker often attempts to exhaust the network or server resources of the intended victim. According to Sisalem et al. (2006), exploitable SIP resources include memory, CPU processing power and network bandwidth. Similarly to memory or CPU time, the end-users can be considered as an exploitable and exhaustible resource. When the end-users are bombarded with spurious calls, they cease to answer to even legitimate calls.

4.1.1 Memory

The processing of SIP messages in a proxy requires incoming requests and replies to be copied into the server's internal buffers. The amount of buffered data and how long it is stored depends on whether the proxy is working in a stateless or stateful mode. At least while the proxy is communicating with an external entity such as a DNS or location server, the data must be kept.

The size of a SIP message varies from a few hundred bytes to a few thousand bytes. Stateful proxies might need to save the state information for 30 seconds or even longer (Ehlert et al., 2008b). Thus flooding a stateful proxy with a stream of messages having different session identifiers will drain its memory very quickly.

4.1.2 CPU

The CPU resources are needed for parsing the received message, transaction mapping, message processing and forwarding the message or replying to it. The message content and type dictate the actual CPU time consumed. In order to authenticate a user, a SIP server generates a nonce value to which the user responds. The user credentials contained in the response message are matched against the user information retrieved from a database. Further, this verification process involves a calculation where a hashing scheme (e.g. Message-Digest algorithm 5, MD5) is used. All these steps consume processing time.

Ideally, a proxy should be sized in a way that it is able to process all messages up to the maximum capacity the link has. However, there are server operations that can be exploited to block the server. In addition to processing needed for user authentication, a SIP server might need to launch a certain application whenever a request is received. An attacker could abuse this feature by registering himself as a legal user and making sure a complex script is executed on a server every time the server receives a request destined for this user.

4.1.3 Bandwidth

A network bandwidth is the maximum transfer capacity of a link needed by a SIP entity for sending and receiving of messages. Attackers aim to overload these links by sending large amounts of useless messages to the SIP end points. When legitimate SIP messages are lost due to network congestion, session setups take longer or fail completely. In the case of a VoIP call, voice quality degrades or the call gets disconnected. Besides hindering SIP traffic, every other service using the same access link suffers from the attack, too.

4.2 Attack Variations

Adapted from Ehlert et al. (2008a), intentional SIP specific attacks are classified into the following major categories:

- Flooding attacks
- Signaling attacks
- Malformed messages

The original categories were “Flooding attacks”, “Malformed messages” and “Irresolvable DNS attacks”, but the author considered the latter to belong to the flooding category. Further, Geneiatakis and Lambrinoudakis (2007) discuss “Signaling attacks” which deserved a category of their own.

With reference to the above categories of deliberate attacks, Sisalem et al. (2006) remind about unintentional DoS attack potential. Usually, these attacks originate from poorly implemented SIP UAs. Although these kinds of attacks are not as massive as “real” attacks, they take place more frequently making them a notable threat.

The SIP protocol can be attacked also indirectly by targeting its transport protocols (see e.g. “SYN flood” in Section 2.3.1) or by tampering with the DNS entries used by the SIP end points.

4.2.1 Flooding Attacks

According to Ormazabal et al. (2008), the most significant attacks in this category use massive volumes of INVITE or REGISTER messages destined for a SIP proxy and intended for exhausting its resources. Although flooding is the most basic type of a DoS attack, at the same time it is arguably the hardest to handle. The SIP flooding attacks can originate from a single source, but distributed (DDoS) attacks pose the most prominent threat to the SIP end points. Again, as already indicated in Section 4.1.1, stateful proxies are more vulnerable to flooding attacks than stateless ones.

When the message flooding is done on top of UDP, an attacker is able to hide his tracks as the source address can be spoofed. If the attacker chooses to use TCP as the transport protocol for SIP messages, the author believes that IP spoofing is practically impossible. This is due to the TCP handshake being performed before the message is processed at

the application level. In order to succeed, the attacker should be able to receive the response packets from the target which is very hard to accomplish.

The Amplification Effect

One possibility for message flooding is to use an amplification effect provided by the forking proxies. It is very beneficial for an attacker to seek for the amplification effect whenever possible because when amplified the magnitude of an attack grows (measured either in quantity or size of packets).

As described in (RFC 3261), an attacker could include forged Route header field values containing victim's address in INVITE requests sent to a forking proxy. Provided that the dummy recipient set in the To header field has multiple contact addresses registered, the proxy will send all forked messages to the intended target. Alternatively, an attacker can register a dummy AoR having multiple contact addresses designating the same (victim) host and use this AoR in an INVITE request sent to a forking proxy. Also, as mentioned by Ormazabal et al. (2008), using multicast to send SIP messages greatly increases the potential for attack amplification.

Another type of attack using the amplification effect is called as the “voice hammer” described in (RFC 4732). We recall from Section 3.4.2 that SIP uses the SDP protocol for carrying address information for the media recipient. An attacker just has to set the victim's IP address within the SDP payload and send INVITE request to a high bandwidth media provider, for instance, to a streaming video server. This way a single request of a modest size could result in a stream of media up to megabits per second. These amplification attacks described above can be classified as reflection attacks as well, since they use innocent third parties to relay the attack traffic (see also Section 2.3.2).

The SIP specification mentions the possibility of infinite redirection loops when two locations are configured to redirect invitations to each other. If the same Request-URI received from a redirect server in a 3xx class response is used more than once, a UA faces an infinite redirection loop (see Figure 10 for illustration). This is also a special case of the amplification effect not involving flooding.

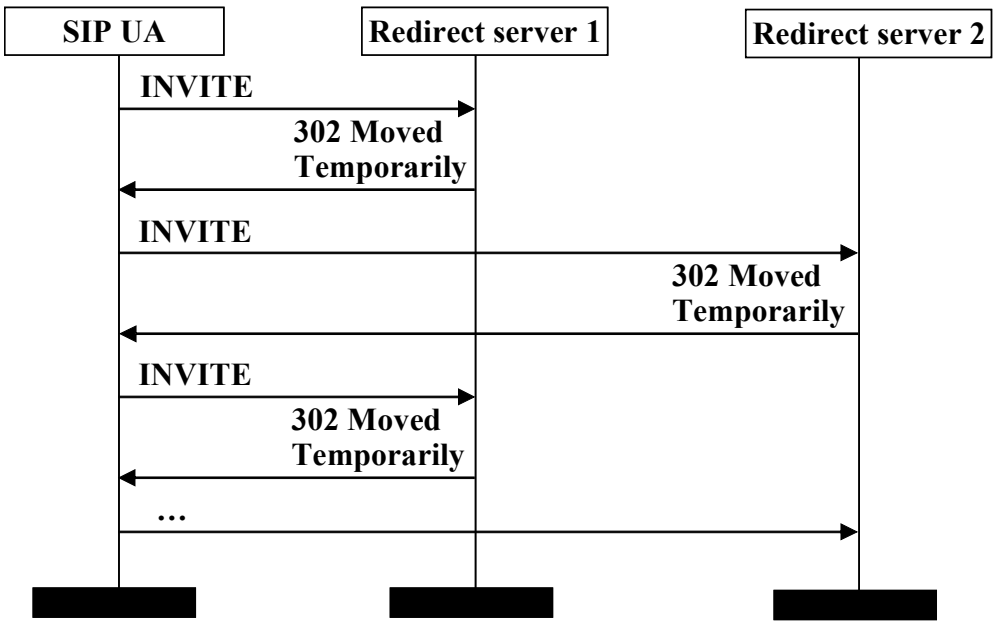

Figure 10: Infinite Redirection Loop

As a variation of the attack using a forking proxy, an attacker could register only one contact address designating the victim's IP address. For this kind of a DoS attack to succeed, the attacker should be able to entice a mass of people to call his (previously unknown) AoR. One method to lure people to call could be masquerading as a help desk for a popular product (e.g. iPhone) and advertising the fake AoR all over the Internet.

DNS Flooding

Zhang et al. (2007) consider DNS to have a major role in a SIP network. Indeed, a SIP message can contain FQDN addresses in headers such as Via, Route, Contact and Request-URI. Some DNS attack possibilities were discussed earlier (a direct attack against a DNS server or tampering of the DNS records) in this thesis, but there is one additional, rather simple and effective way to degrade the performance of the SIP infrastructure. The method in question is to disturb SIP message processing by inserting unresolvable host names into a SIP message.

Once a SIP proxy encounters FQDN address in a header field used for routing the message (e.g. Via), it must query the local DNS cache for getting a corresponding IP address mapping. If the needed address is not in the cache, an authoritative name server for the domain in question has to be queried. According to Jung et al. (2002), in a normal case getting an answer from DNS takes an average of 1.3 queries with a median

latency around 100 ms, but configuration errors could cause substantially higher figures. However, the attacker is able to take the aforementioned numbers to a whole new level by using randomly generated host names of a domain whose authoritative name server is known to have a poor response time. Provided that the attacker has selected the FQDN addresses well, no results can be produced until a timeout (5 seconds in many installations) is reached at the DNS subsystem.

While the proxy is waiting for an answer from the local DNS server, some of its resources are blocked. If the proxy is operating synchronously and does not have parallel processing queues implemented, its CPU will be totally blocked during this time, otherwise state information will eventually exhaust its memory. In fact, Zhang et al. (2007) have found out that 8 GB of Random Access Memory (RAM) can be depleted in 30 seconds or so with a DNS flooding attack. The anatomy of the DNS flooding attack is shown in Figure 11.

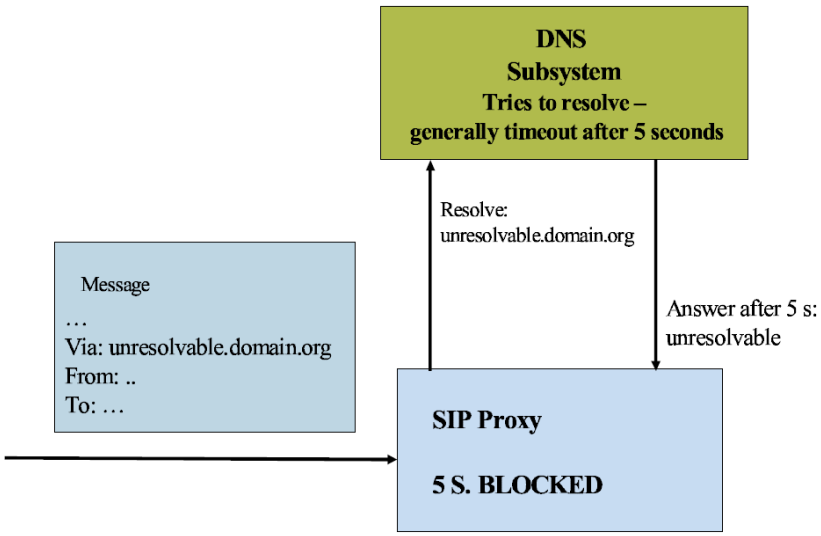

Figure 11: DNS Flooding Attack Using Unresolvable Host Names (Source: Zhang et al., 2007)

Since the attacker is using SIP messages that comply with the SIP standard, it makes it difficult to filter these messages either in the SIP proxy or in the Intrusion Prevention System (IPS). From the author's point of view, this aspect makes the DNS flooding attack a very serious threat. Moreover, it should be noted that in this case even a stateless proxy is at risk.

4.2.2 Signaling Attacks

This category of attacks exploits application-level vulnerabilities by manipulating features of the SIP protocol syntax to cause a DoS condition at the target. If REGISTER requests are not authenticated, this enables a range of DoS attack possibilities presented in (RFC 3261) and (Cha et al., 2008). An attacker could exhaust memory and disk resources of a registrar by registering a large number of contact addresses. In addition, an attacker can prevent the target user from receiving calls by canceling an active registration. This is done by sending a forged REGISTER request containing Expires header field set to 0 (see Figure 12). Otherwise, a registration can be hijacked by falsifying contact address to be that of the attacker.

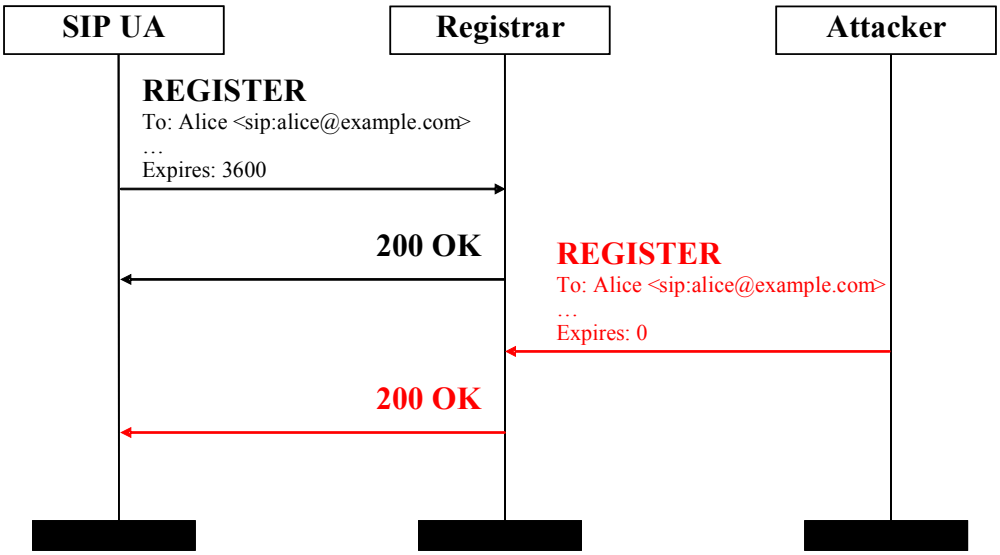

Figure 12: De-Registration Attack

The call hijacking explained in Ormazabal et al. (2008) is feasible by injecting a "302 Moved Temporarily" message to an ongoing SIP dialog. Alternatively, an attacker can merely redirect media streams to an unauthorized address and thereby wiretap the dialog by sending a spoofed re-INVITE request with modified session description parameters in the SDP payload.

Geneiatakis and Lambrinoudakis (2007) present the BYE attack depicted in Figure 13. The BYE attack is accomplished by sending a spoofed BYE request that illegally terminates an active session. In the message sequence shown below, the attacker is impersonating "UA 2" when sending the BYE message to the unsuspecting recipient ("UA 1").

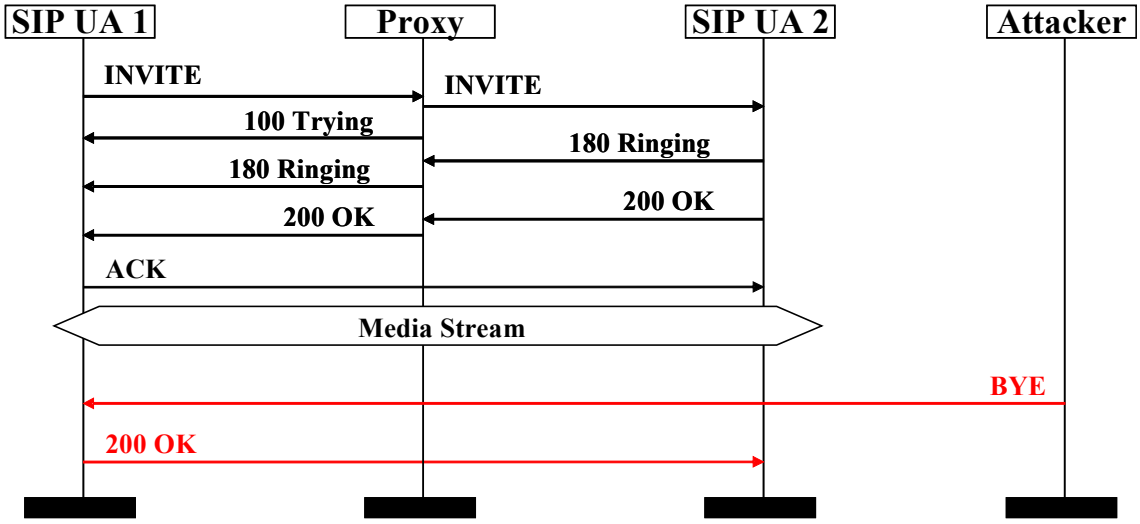

Figure 13: Hostile Call Termination

The CANCEL attack can be performed similarly. Whereas the BYE attack can commence any time after the session is set up, the CANCEL attack must be completed before the caller receives a final response (e.g. "200 OK") from the callee. In the example shown in Figure 14[10], an attacker sends a CANCEL request to a proxy mimicking "UA 1", but the spoofed message could also be sent directly to "UA 2" for the attack to be successful.

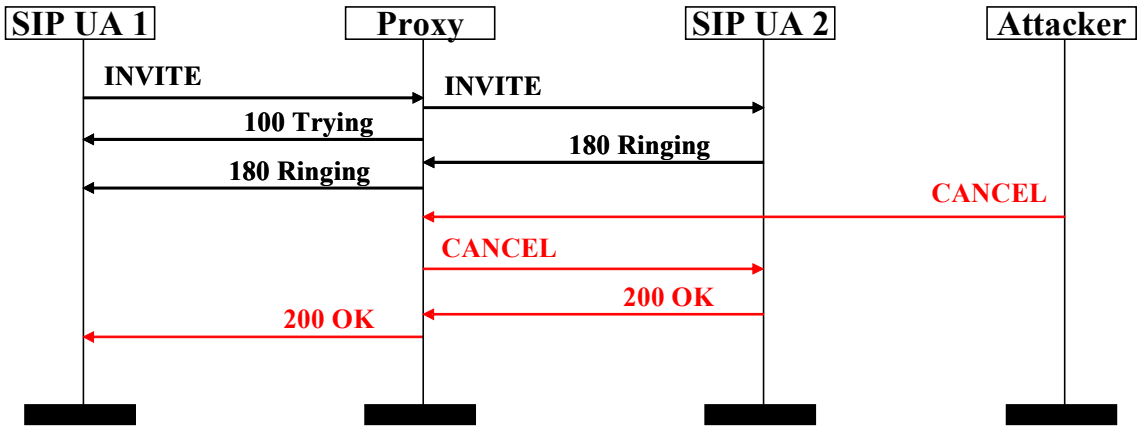

Figure 14: CANCEL Attack

As opposed to flooding attacks, before launching a signaling attack an attacker needs to acquire the session identifiers (including the Call-ID, CSeq, the tag in From header and in

[10] The message sequence in Figure 14 is adapted from (Cha et al., 2008) as the author considered the original chart to contain errors.

some cases the tag in To header) for the attack to be successful[11]. This is done by eavesdropping the target users' session establishment messages. (Geneiatakis and Lambrinoudakis, 2007)

4.2.3 Malformed Messages

As mentioned earlier, CPU resources are needed for parsing the incoming messages. In effect, a SIP proxy must parse at least part of the message and perform a consistency check on it. As a result of the flexible text format allowed by the SIP standard, a message can be compliant with the standard although it is intentionally assembled to hinder the parsing process. However, any flaws in the implementation of a SIP entity offers a channel for attacks. A faulty SIP implementation can be exploited by sending a carefully crafted packet causing an excessive resource usage or even a system crash or reboot (Ormazabal et al., 2008). Besides, SIP parsers typically expect to receive well-formed messages as pointed out by Ehlert et al. (2008a).

Tormenting the Parser

Sisalem et al. (2006) present several possibilities that an attacker can use to complicate the parsing process. To begin with, the attacker can construct unusually long messages using additional informative header fields such as Supported or Allow. These header fields can be used together with a large message body albeit a body is not needed in every message. A well implemented parser might ignore unknown headers, so the most effective way is to use only header fields listed in (RFC 3261). A longer message consumes more processing power, memory and network bandwidth. Next, it is possible to include a message body whose size does not match with the value in the Content-Length header field in order to crash the parser.

Additional methods to hamper the parsing process include spreading headers of the same field all over the message as discussed in Sisalem et al. (2006). The SIP standard allows the distribution of certain headers that have multiple values into individual fields containing only one value. For example, the following headers can be separated into individual header fields: Accept-Language, Allow, Contact, Route, Supported and Via. To illustrate the possibilities, three alternatives to distribute multiple Contact fields are shown in Figure 15.

[11] Session identifiers are not necessary for REGISTER attacks.

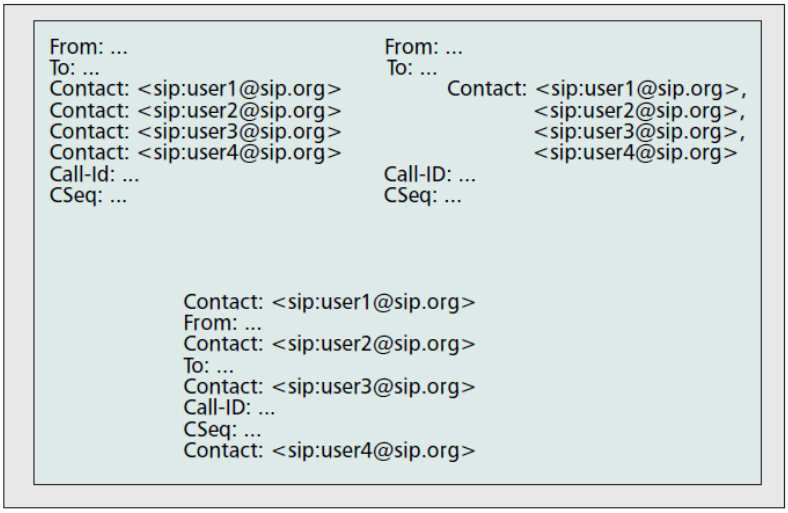

Figure 15: Various Possibilities to Distribute Headers (Source: Sisalem et al., 2006)

Further, to complicate the parsing even more, standard compliant SIP entities must be able to process RFC 2543[12] messages. If the branch parameter in a Via header does not begin with a magic cookie "z9hG4bK", the received message is to be handled according to the older and more complex rules from RFC 2543. The same applies if To and From header fields do not contain tags. In consequence, the proxy needs to match the received message against all active transactions wasting valuable CPU time in the process.

Sisalem et al. (2006) also suggest that it would require more processing power to parse messages that have headers containing routing information (e.g. Via and Route) placed in the end of the message. The author disagrees as the message must be parsed completely anyway (the reader is referred to the discussion in this section about the possibility of spreading the headers).

Naturally, the attacker can also send messages that clearly violate the SIP standard to cause unexpected behavior at the receiving end. For example, the INVITE message shown in Figure 16 is syntactically incorrect as it lacks a mandatory Request-URI in the Start line of the message (Ehlert et al., 2008a).

[12] Original version of the SIP standard now made obsolete by RFC 3261.

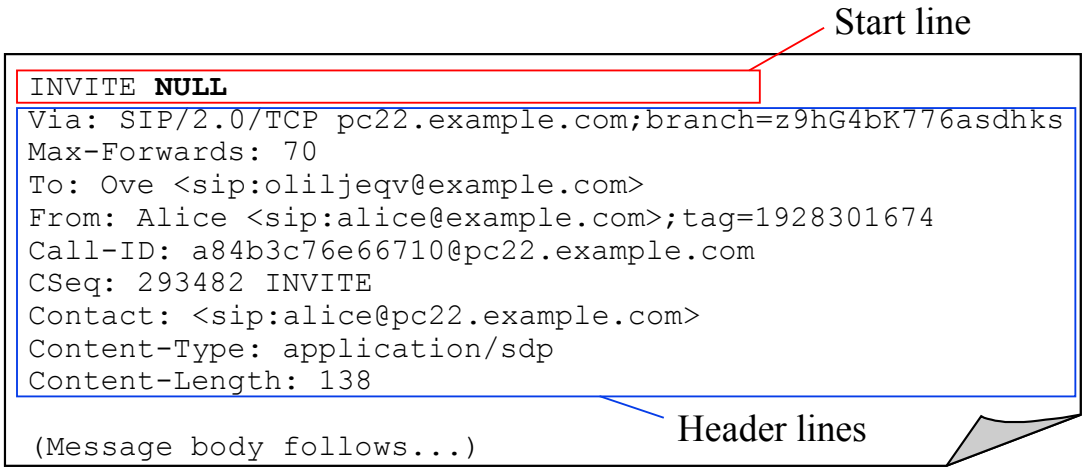

Figure 16: Malformed INVITE Request

SQL Injection

In addition to direct attacks on the parser, the supporting database services of a proxy can be targeted. Ehlert et al. (2008a) show an attack method where an attacker utilizes Structured Query Language (SQL) injection for sending malicious SQL code within an Authorization header (see the example in Figure 17). This kind of attack can be used every time a proxy is asking for authentication, given that the authentication information is stored in a relational database instead of a LDAP directory, for example.

```
Authorization: Digest username="alice';
      Update subscriber set first_name='malicious'
      where username='alice'--",
      realm="example.com", algorithm="md5",
      nonce="84a4cc6f3082121f32b42a2187831a9e",
      response="7587245234b3434cc3412213e5f113a5432"
```

Figure 17: SQL Injection Attempt

The malicious code demonstrated in the example is embedded in the username field, but could reside also in the realm field of the Authorization header. When the SQL query is run to verify the received credentials, the attacker hopes that the *Update* command he injected would be run as well. Even though the authentication will fail in any case as the attacker does not have a valid password, the attack might manage to change the first name of the user "alice" to "malicious". In short, using similar messages the attacker could corrupt the database preventing users from authenticating.

4.3 Summary

This chapter described the various ways a DoS attack can be mounted against SIP. A malicious user can exploit the same common vulnerabilities associated with any Internet service, but SIP has its own vulnerabilities that can be taken advantage of. Furthermore, exhaustible SIP resources (CPU processing power, memory and bandwidth) were introduced. We focused on attacks against a SIP proxy since it is a valuable asset of a service provider but vulnerable as it is exposed to the Internet.

The protocol specific attacks included flooding which aims to deplete the resources needed by the SIP entities and signaling attacks that exploit protocol features. Also, the attacks using malformed messages and targeting the implementation flaws were discussed. Among these attacks, the message flooding was found to be the most difficult to handle. The attacks described included a simple yet devastating DNS flooding attack using unresolvable host names and flooding attacks using the amplification effect.

As opposed to the flooding attacks, most signaling attacks require that the attacker is able to eavesdrop the target users' session establishment messages and acquire the session identifiers for the attack to be successful. Because the attacker needs to gather more information to initiate a signaling attack these attacks pose a smaller threat than the flooding attacks.

We mentioned that the attacks with malformed messages aim to cause an excessive resource usage or even a system crash. Typically, these attacks reach their goals by hampering the parsing process. Examples of these attacks included the use of unusually long and syntactically incorrect messages.

In conclusion, the author has come up with one especially threatening attack (sort of an “All-Star” attack): A malicious user could combine various properties from the attack techniques described in this chapter. For instance, he could use DNS flooding with messages having headers of the same field spread all over the message while the message syntax followed an older RFC 2543 standard.

5 DoS Protection Mechanisms for SIP

Denial of Service attacks are much harder to defend against than other invasive attacks trying to take over the target. Also, they are impossible to eliminate in practice (Ormazabal et al., 2008). Therefore, it is important to design and implement innovative solutions targeting the threat that DoS attacks pose to SIP.

This chapter investigates different ways of preventing and mitigating the effects of Denial of Service (DoS) attacks on SIP services. Although some application independent defenses are discussed and evaluated, this thesis focuses on SIP-specific protection possibilities. Similarly as in the previous chapter, the focus is on protection for the SIP proxies when end points are considered. Further, the solutions presented are evaluated from a service provider's point of view.

5.1 General Protection Mechanisms

Before applying SIP specific defense mechanisms, it is worthwhile to implement generic protection for the whole infrastructure (i.e. mechanisms that are not application specific). Obviously, as stated by Peng et al. (2007), by making huge investments in server and network capacities, it is possible to eliminate most if not all DoS attacks. However, as this strategy is only available for large corporations such as Microsoft, the rest of us need to be more innovative.

5.1.1 Perimeter Protection

All software components including operating systems should be kept up-to-date by patching them regularly as recommended by Lau et al. (2000). In addition, a server hardening by disabling unused services and using a vendor recommended secure configuration should be considered as a necessity these days.

As we are trying to maximize the total availability figure given by Eq. (3) in Section 2.1, critical infrastructure servers and network components within the service provider's local network should be overprovisioned to increase their availability. Further, different servers (especially SIP proxies) should be put behind a load balanced firewall cluster in their own network segment. This way it would be possible to curtail the collateral damage to other services when some particular service is under attack.

While technically outside the local network, Internet access links are very much under the influence of the service provider. In fact, the availability of the Internet connection can be increased using multiple Internet Service Provider (ISP) links, similarly as server clustering improves server availability.

Eddy (2006) presents various protection alternatives against SYN flooding. One viable method is to use SYN cookies to eliminate the need to store any state information for the received connection request (TCP SYN message). In effect, even a large flood of SYN messages would not waste any server resources (though bandwidth would still be consumed). SYN flooding protection should be implemented at a firewall or load balancer level.

Intrusion Detection and Prevention Systems

A general purpose Intrusion Prevention System (IPS) monitoring for malicious traffic in a network and blocking the offending packets when needed is an extra layer of protection against DoS attacks. According to Peng et al. (2007), IPS or an Intrusion Detection System (IDS) commonly use either signature-based or anomaly-based techniques to detect attacks.

Signature-based detection relies on known patterns of malicious activity. However, this method is in practice ineffective against DoS attacks since attackers can easily vary the type and content of the traffic making it difficult to implement accurate signatures. On the other hand, anomaly-based detection is able to spot an attack if it does not match the traffic profile generated during a so called training period. A challenge with anomaly-based detection is that it is difficult to minimize *false positives*[13].

Anomaly-based detection can be effective against the type of DoS attacks involving a large number of packets from a relatively small number of sources. In contrast, it is hard for the anomaly-based technique to distinguish DDoS attacks mimicking legitimate traffic from *flash crowds* where genuine users access the service at the same time.

[13] A false positive occurs when legitimate traffic is classified as an attack

5.1.2 Defenses Outside the Local Network

The closer the defenses are deployed to the attack source, the more effective they are in reducing the damage to the service offered. Unfortunately a service provider is usually limited to using merely local defenses (see previous section). In the case of a massive, highly distributed DDoS attack, perimeter defenses are not sufficient. Even redundant network access links are eventually saturated under huge traffic volumes and legitimate traffic is not able to reach the service.

Router-Based Methods

One of the techniques to combat DoS attacks closer to their sources is to use *ingress filtering* described in (Peng et al., 2007). Ingress filtering is a packet filtering scheme used in ISP routers for making sure that only packets with valid (non-spoofed) source IP addresses are let through. This decision is made according to the expected IP address range inside the ISP network. IP spoofing would still be possible within the IP range of a particular ISP, but at least the attack source could be traced on an ISP level. Nevertheless, for this scheme to be effective, it would require worldwide adoption among thousands of ISPs and as noted earlier, DDoS attacks do not need to use IP spoofing to be powerful.

Other similar packet filtering techniques presented in (Peng et al., 2007) include Router-based Packet Filtering (RPF) and the Source Address Validity Enforcement (SAVE) protocol. RPF is basically the same as ingress filtering, but on an Autonomous System (AS) level. In a routing sense, ASs form the core of the Internet. RPF requires lesser adoption (roughly 2000 ASs) than ingress filtering to be effective, but then again it needs modifications to the BGP routing protocol. The SAVE protocol has update functionality so that routers are able to periodically refresh information of expected IP address ranges on each link. As SAVE is a new protocol that has to be deployed at routers, it will take a long time for it to make any difference.

In addition to filtering, it is important to be able to trace attacks to their sources so that attackers can be prosecuted. Peng et al. (2007) review a promising source identification technique known as Hash-Based IP Traceback. In this scheme routers store information on all packets they forward. If a traceback is needed, the target of the attack sends a query for any attack packet to its upstream routers and eventually the source is found. Obviously, the coverage of routers having a traceback functionality dictates how

successful this method is in finding the actual source. Besides, storing information on forwarded packets is an enormous overhead in high-speed routing environments.

All these routing related schemes require almost universal adoption and are only effective against DoS attacks using spoofed IP addresses. These methods are outside the control of a service provider and ISPs and ASs lack the motivation to take these kinds of techniques into use. Moreover, DDoS attacks with armies of zombie hosts using non-spoofed addresses remain as an unsolved problem. Security awareness among ordinary Internet users should be increased and ISPs would be in an ideal position for educating their customers and offering them security software at attractive prices. This way botnet sizes and their attack power could be decreased.

Other Methods

Handley and Greenhalgh (2004) have made an interesting proposal for a new Internet architecture. In their model IP address space is divided into a client and server addresses. The idea is that clients can start connections to servers, but not to other clients. In addition, servers are not allowed to initiate connections to other servers. These restrictions would mean that worms and viruses could not spread as quickly as they now can, reducing DoS threats that stem from worm or virus infections. Also, due to other routing characteristics in their model, IP spoofing would not be possible.

In this model, CPU puzzles[14] can be used before a client is allowed to setup a connection to the server. The overhead caused by this verification process is negligible for a normal client, but enough to constrain the rate at which a malicious user can initiate connections. In addition, special-purpose firewalls are deployed at inter-domain boundaries to throttle connection requests in case a server under attack asks for help.

Arguably the proposed architecture would be significantly more robust against DoS attacks than the current Internet architecture. However, changes to the current architecture are so radical, that the model will not receive any global acceptance in the near future.

[14] CPU puzzle is a calculation that is hard (CPU intensive) for the recipient to solve, but easy for the sender to check.

5.2 SIP-Specific Protection Mechanisms

To defend the service against the most sophisticated attackers, application specific protection measures must be utilized. Next, we will cover countermeasures that can be used for protecting SIP services against different DoS attack variations.

5.2.1 The Importance of Authentication

Protection against resource exhaustion attacks starts by authenticating all requests to a SIP proxy (Sisalem et al., 2006). The SIP protocol uses digest authentication (see Section 3.7.1), but as it needs to store a state in a memory, digest authentication can be exploited in a memory exhaustion attack. Hence, predictive nonces (Rosenberg, 2001) should be used to achieve stateless authentication. In this scheme, a nonce value is calculated over specific headers that do not change from the original request. The use of predictive nonces is also supported by the fact that no changes to the SIP protocol are needed. Furthermore, this approach provides integrity for a set of SIP headers (e.g. To, From and Contact). Authentication is also essential in defending against DNS flooding attack (see next section).

It is not trivial how users register to use services. If user identities are not verified at all, the benefit from using the request authentication is lost. One solution would be to use a credit card in the registration[15], since credit cards have worldwide acceptance.

Sisalem et al. (2006) recommend maintaining a list of suspicious users at a SIP proxy. This way misbehaving users could be blocked or their ability to establish sessions could be limited. Again, authentication is crucial in ensuring that the listed users are real and not generated by an attacker. In addition, a proxy should execute as many checks as possible in a stateless mode.

5.2.2 Flooding Attacks

It was pointed out in the previous chapter that the message flooding is the most difficult DoS attack type to handle. Fortunately, there are reference implementations available to combat the flooding attacks. One of these implementations is known as “Secure SIP” by Ormazabal et al. (2008).

[15] This verification process could use similar methods as PayPal (an online payment company), but details are beyond the scope of this thesis.

Secure SIP

In essence, Secure SIP is a SIP-aware application-layer firewall suitable for high-speed networks. This firewall is designed to filter SIP traffic in two phases. Media traffic (RTP streams) is allowed through Filter 1 according to the SDP session information, whereas Filter 2 handles the SIP signaling traffic (i.e. traffic destined to standard SIP port 5060) using a series of SIP-based filters to protect the actual SIP proxy. All other packets are dropped at Filter 1. The whole process is illustrated in Figure 18.

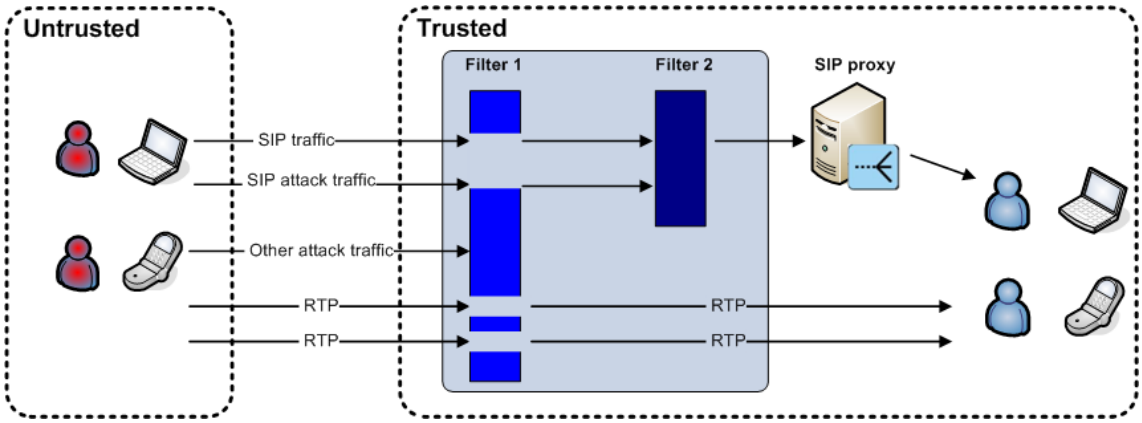

Figure 18: Secure SIP Solution Overview

These SIP-based filters include a return routability filter, a rate-limiting filter and a state-validation filter. All the filters are hardware-based to achieve high performance. The return routability filter is needed against spoofed UDP packets. We recall from the previous chapter that the spoofing of TCP packets is practically impossible on the application level, although the SYN flooding has to be handled somewhere. Consequently, this solution could be augmented by adding a SYN flooding prevention logic to Filter 1. Here, the return routability filter is based on digest authentication leveraging "null authentication" that allows anonymous users with no passwords. Although this is a clever solution against spoofing in a testing environment, it would be better to use predictive nonces allowing stateless operation (not to mention that anonymous users should be blocked in a production environment).

The other remaining filters, the rate-limiting filter and the state-validation filter, work together in preventing SIP message flooding with real IP addresses (now that the risk from IP spoofing is greatly reduced). Rate-limiting can be done on both SIP transaction and dialog level as the firewall constructs the corresponding dialog and transaction

identifiers and maintains states following the SIP state machine as specified in the SIP standard. For instance, floods of INVITE messages having the same transaction identifier can be filtered.

In the tests by Ormazabal et al. (2008), the unprotected SIP proxy collapsed with fewer than 200 spoofed attempts per second, whereas Secure SIP was able to handle over 14,000 spoofed requests per second (the used testbed could not generate more traffic). Still, it is unclear to the author whether all these spoofed requests had unique source IP addresses. The return routability filter is effective against sequential spoofed requests from a single source, but is likely to be vulnerable to attacks where high volumes of SIP messages with unique IPs are used. A Secure SIP firewall is also dependent on the SIP proxy to give filtering orders to the return routability filter.

VoIP Defender

Fiedler et al. (2007) have designed an open security architecture called VoIP Defender with a focus on flooding attacks. This architecture is designed to be scalable and extendable with new detection algorithms. Further, packet analysis and control is possible in all layers from the application layer down to the link layer (the reader is referred to Figure 2 in Section 3.4). In contrast to Secure SIP, VoIP Defender acts as a traffic repeater and, therefore, does not change the packets it handles. Hence, it is completely transparent for the UAs and the proxy it protects.

Ehlert et al. (2008b) have extended this architecture to include a similar state machine as the one used in Secure SIP. Operating as a VoIP Defender module, the state machine gathers statistical data from the SIP message flow and is able to detect suspicious traffic based on the collected data. Measurements for the attack detection are done on three levels: transaction, sender (based on source IP) and global level. The global level is a statistical summary of all recent transactions. For instance, contrary to Secure SIP, VoIP Defender is able to detect an attack that uses different transaction identifiers within the same dialog. Even flash crowds are separable from the attack traffic by analyzing state information, for example, processing times and timeouts.

To give an idea of the performance, VoIP defender was able to process over 50,000 requests per second in a testbed with IP-based rules. As a comparison, the state machine was able to process 2,800 requests per second, but the test setups were different. Also, it

would require an identical testbed to compare the actual performances of Secure SIP and VoIP Defender. Both solutions should be fine-tuned before placing them in a production environment to have an optimum fit to the target network conditions.

DNS Flooding

In the previous chapter, DNS flooding was found to be a simple, yet effective DoS attack. Therefore, it deserves a protection mechanism of its own. One such method is the layered defense architecture proposed by Ehlert et al. (2008a), where DNS flooding protection is a single component.

In their experiment, Ehlert et al. (2008a) found out that a dedicated DNS cache implementation in a SIP proxy with parallel message processing capabilities is needed against the DNS flooding attacks. As discussed in Section 3.6.2, a SIP endpoint uses NAPTR and SRV records for locating a SIP server. These DNS records need to be cached for an optimum performance and a general operating system DNS cache does not cache these records. In addition, this cache needs a specialized cache replacement policy and the Least Frequently Used (LFU)[16] cache algorithm was found to yield the best results.

The proxy is able to conclude that there is an ongoing DNS flooding attack, if almost all its message processing queues are concurrently resolving DNS names. For example, if there are 16 queues, it can be defined that as soon as the 15th queue is in use, the proxy stops issuing new DNS queries and uses only cached content for replies. This way the current connections and requests to popular destinations can be served normally. It is important to note that only successful queries are cached. Otherwise the cache would be filled with irresolvable entries.

Although insufficient alone, a DNS timeout value mentioned in Section 4.2.1 could be reduced to further mitigate the effects of a DNS flooding attack. Further, Sisalem et al. (2006) recommend to ignore the topmost Via header in a SIP message in case it contains a FQDN address and to use the packet's source IP address instead to reduce a burden to the DNS service.

[16] In the LFU cache replacement policy, the DNS usage frequency is measured and cache entries are ordered by their frequencies. The entries that are least used are discarded first from the cache.

The Amplification Effect

Many of the amplification attacks presented in Section 4.2.1 used forking proxies as tools for increasing the attack power. Therefore, a service provider should decide whether parallel forking capabilities are really needed or should these capabilities be limited only to its own, perhaps more trusted, customers. Also, it is a good idea to limit the number of contact addresses a user can register.

The “voice hammer” attack is particularly tricky to prevent. One solution would be to change the SIP protocol to include a some kind of a handshake where it is determined whether the target host wishes to receive the media stream before it is sent. Otherwise, the entity fearing this attack can hide behind a firewall that opens RTP ports dynamically (e.g. Secure SIP). Nevertheless, the access link is flooded in the latter option.

5.2.3 Signaling Attacks

As described in Section 4.2.2, most of the signaling attacks rely on the attacker's ability to eavesdrop the target users' session establishment messages. Consequently, a natural solution is to use encryption to ensure the confidentiality of the communication. Depending on the transport protocol, either TLS or DTLS could be used. Alternatively, IPSec or S/MIME can be utilized, but their efficient usage requires the presence of a PKI. See Section 3.7.1 for more details on these encryption methods.

If the use of encryption is not feasible, Geneiatakis and Lambrinoudakis (2007) have proposed a new header for SIP called Integrity-Auth to defend against signaling attacks. This header provides both message authentication and integrity by using hashing which introduces a small computational overhead. Moreover, the increase in the message size is negligible (the actual increase depends on the used hashing algorithm).

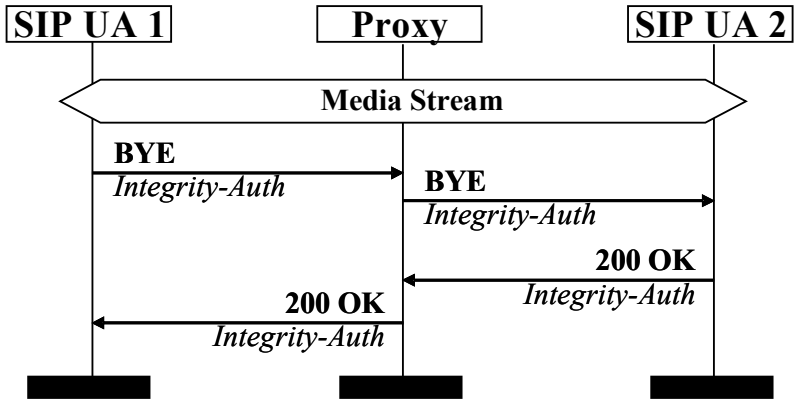

Figure 19: Using Integrity-Auth Header with the BYE Method

Geneiatakis and Lambrinoudakis (2007) argue that the Integrity-Auth header protects from any kind of a signaling attack and is not vulnerable to the man-in-the-middle attacks, but they do not support these claims by presenting any test results. Also, it should be noted that in order to protect from the BYE attack, a proxy must remain on the signaling path[17] (see Figure 19) because it is the one that validates the identities of the end users. In addition, as Geneiatakis and Lambrinoudakis (2007) do not discuss a case where UAs belong to the different domains, it is not clear to the author whether this scheme would work in that case. At least a mutual authentication between proxies of the different domains would have to be arranged.

Another protection scheme to combat a specific set of signaling attacks has been presented by Cha et al. (2008). Their scheme involves a new SIP method called RETRANS where a SIP server, typically a proxy, asks a UA to resend the most recent SIP message if it was either CANCEL, BYE or REGISTER (with the Expires header set to "0", i.e. registration cancellation). In case the server receives only a single response which is identical to the message that was received earlier, the message processing continues normally. Otherwise the server concludes there is an ongoing attack and alerts the UA (see Figure 20). Similarly as when using the Integrity-Auth header, the proxy must stay on the signaling path to protect against the BYE attack.

[17] As can be seen from Figure 5, in some cases after the call establishment, the signaling continues directly between the UAs.

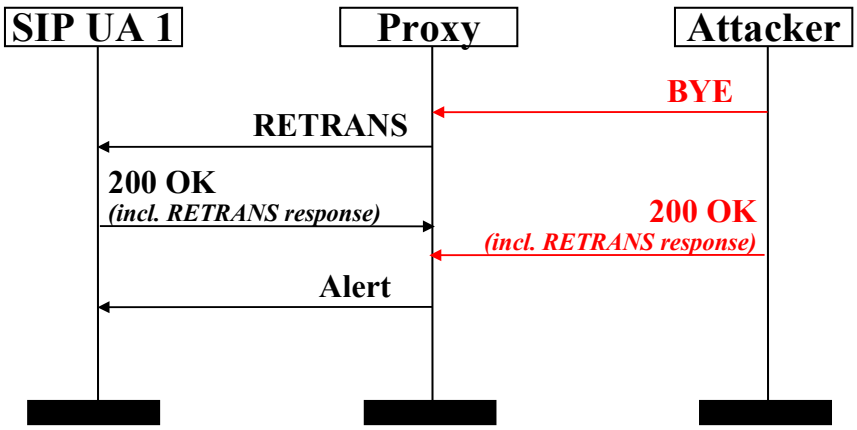

Figure 20: Preventing an Attack with the RETRANS Method

There are multiple limitations with the proposed RETRANS method. To begin with, it does not protect from the man-in-the-middle attacks. Also, if compared to the Integrity-Auth header, the RETRANS method introduces more message overhead and is more complex to implement and support. Further, it would be safer if the REGISTER requests were required for retransmission if the Expires header contained a suspiciously low value (e.g. 1800 seconds or less). For example, a clever attacker could set the expiration time to one second instead of a zero.

By now it is clear that all requests should be authenticated in one way or another. For instance, the author believes that the simplest and most effective way to secure the registration process is to authenticate users using predictive nonces. If the predictive nonces are used in authentication, even the man-in-the-middle attacks related to registration (e.g. hijacking or canceling a registration) can be prevented.

5.2.4 Malformed Messages

To protect a SIP implementation from the DoS attacks, Sisalem et al. (2006) advise that developers should use fast and efficient mechanisms for message parsing and memory allocation. Further, the implementations should decide on a size limit for messages and reject messages exceeding this limit. Also, in case there is a message body, the value in the Content-Length header field should not be trusted to match the actual size of the body.

As pointed out in Section 4.2.3, the SIP standard mandates that the SIP entities must be able to process RFC 2543 messages. In the author's opinion, it would be beneficial for a service provider to determine the percentage of its clients not compliant with the current

standard. In case the percentage is not significant, a heavily loaded SIP proxy could temporarily switch off the support for RFC 2543 messages until the load returns to normal.

Ehlert et al. (2008a) also have the protection for malformed messages in their layered defense architecture. In the proposed architecture, malformed messages are detected using message signatures (illustrated in Figure 21) based on the grammar defined in the SIP standard. For instance, the SQL injection attack depicted in Section 4.2.3 can be prevented by applying signatures for the SIP headers.

```
SIP_METHOD SIP_URI | SIPS_URI MESSAGE_HEADER+
[MESSAGE_BODY]

Additional rules
SIP_METHOD!=NULL
MESSAGE_HEADER!=NULL
size_of(SIP_METHOD)>%constant% e.g. 50 bytes
size_of(MESSAGE_BODY)>%constant%
```

Figure 21: General Structure of the SIP Signature (source: Ehlert et al., 2008a)

The signatures are enforced in a pre-filtering module which forwards the compliant messages to the parser and rejects all the others. As soon as a message is rejected, a record of it sent to the operator console and stored in a database. Hence, the potential attacks can be further analyzed and signatures fine-tuned. Besides, the delay caused by the pre-filtering module is insignificant, being less than 100 µs.

In addition to patching the operation system of the SIP server as indicated in Section 5.1.1, it is even more important to keep the SIP implementation up-to-date. Further, security and interoperability testing (e.g. SIPit events) needs to be conducted to ensure a robust and stable implementation (Sicker and Lookabaugh, 2004). Interoperability testing also helps to reduce the number of defective implementations that can cause unintentional DoS. For software security testing, a good starting point is to use the SIP robustness test suite developed by Wieser et al. (2003), which in turn is based on the PROTOS project (PROTOS).

5.3 Comparison of the Protection Mechanisms

To rate the effectiveness and implementation effort of the set of defense methods discussed in this chapter, these methods are placed in a matrix (see Figure 22). The effectiveness in this context portrays how well a particular method protects from the related DoS attacks. If the related attack has a high probability and severe impact on the service, the resulting risk is considered to be high. Moreover, it is here assumed that the easier the attack is to launch, the higher is the attack's probability.

The implementation effort in turn depicts the overall work needed to deploy the defense mechanism. For instance, if a particular defense method requires changes to the SIP standard, the effort reflects not only the work done by a single service provider, but the affected clients and standardization bodies are taken into account as well. Further, in this context a high implementation cost translates to a high effort.

To sum up, we want to maximize the effectiveness of a countermeasure while keeping the implementation effort low (i.e. the top right corner of the matrix indicates the methods that should be realized first). Also, it should be stressed that both the metrics reflect the author's subjective opinions.

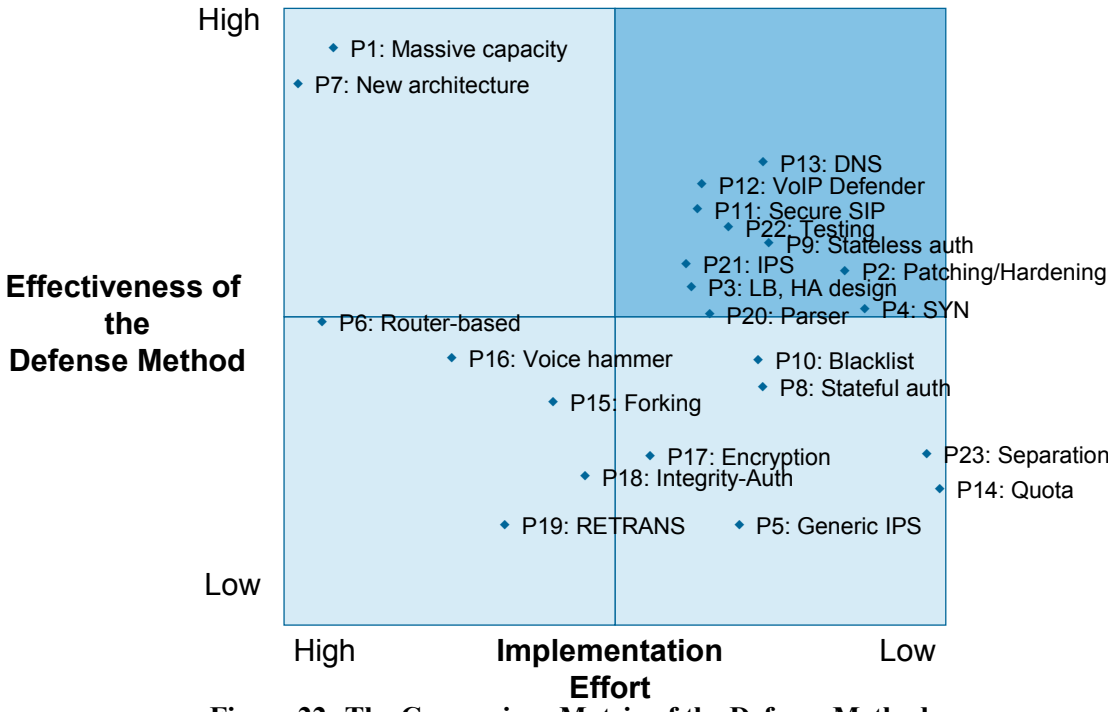

Figure 22: The Comparison Matrix of the Defense Methods

The relative ratings for the defense mechanisms appearing in Figure 22 are briefly explained in the table below. Note that the numbering does not reflect the given rating.

Table 2: Rated Defense Mechanisms

#	DESCRIPTION AND RATIONALE FOR THE RELATIVE RATING
P1	Server and network capacities comparable to the largest enterprises in the world: Very effective, but also very expensive protection method. Still, even a massive capacity is not sufficient alone if a poorly implemented SIP proxy can be crashed with a single packet.
P2	Server hardening and patching (especially the SIP software): Can be considered as a must, relatively low effort needed.
P3	Load balanced infrastructure, redundant servers, ISP links and network components, generic firewall: Serves as the foundation for the other defense mechanisms.
P4	TCP SYN flooding protection: Easy to implement and effective against TCP based attacks with spoofed IPs.
P5	Generic IPS: Almost useless when the system is not SIP-aware.
P6	Router-based methods (ingress filtering, RPF, SAVE and IP traceback): Effective against attacks with spoofed IPs, but it is very difficult to accomplish the needed worldwide adoption among ISPs and ASs.
P7	New Internet architecture: A potent proposition against the DoS attacks, but even harder to get the required universal acceptance than in the previous method (P6).
P8	SIP servers authenticate all requests using normal (stateful) digest authentication: Limited defense against unauthorized users, memory exhaustion possible.
P9	SIP servers authenticate all requests using predictive nonces (stateless authentication): Excellent price-quality ratio, protects also from the man-in-the-middle attacks.
P10	SIP proxy maintains a list of suspicious users: Relatively easy to implement, but users need to be authenticated.
P11	Secure SIP, a SIP-aware application-layer firewall: A promising solution provided that minor improvements are done. Dynamic RTP handling is an advantage, but DDoS might be a problem.
P12	VoIP Defender, a transparent traffic repeater: Operates transparently and independently, effective against DDoS on the application (SIP) level.
P13	A dedicated DNS cache implementation in a SIP proxy: Protects from the DNS flooding attacks that are powerful, but very easy to initiate.

P14	Registration quota: Very low effort to deploy, but a limited defense as well.
P15	Disabling/limiting parallel forking: Effective against certain types of amplification attacks, but could be hard to implement in practice.
P16	Protection from the “voice hammer”: The proposed solution requires changes to the SIP standard, thus it is difficult to accomplish.
P17	The use of encryption (TLS, DTLS, IPSec, S/MIME): Provides a comprehensive protection from the various signaling attacks requiring eavesdropping. However, the threat of eavesdropping is low and the encryption schemes commonly need a PKI environment. Also, encryption and decryption are CPU intensive processes and the use of encryption does not protect from the common flooding attacks.
P18	Integrity-Auth header: A new SIP header does not require major changes to the protocol, but still UAs and proxies would need to be updated to support it. Provides similar protection from the signaling attacks as the previous mechanism (P17), but does not provide confidentiality for the communication.
P19	RETRANS method: Poorly designed defense mechanism for a limited set of signaling attacks.
P20	A well-designed and implemented SIP parser and efficient memory allocation: A parser is in the heart of the SIP implementation, so it is imperative (though not easy) to get it right.
P21	A signature-based, SIP-aware IPS: If done well enough, the parser can be faulty. A good to have defense against the most sophisticated attackers, but needs updating and monitoring.
P22	Security and interoperability testing (SIPit events, PROTOS): The interoperability testing is more geared towards the vendors, but useful for the service providers as well. On the other hand, the security testing is really essential, but obviously needs supportive structures (i.e. good defense methods to be tested). Both types of testing are continuous processes.
P23	Separate proxy and registrar servers: Separation protects the other component in case the other one is under attack.

5.4 Recommendations

Based on the findings in this thesis and the author's own personal experience, the following recommendations to secure a SIP proxy against the DoS attacks can be made:

- Setup a server and network environment with enough capacity. In the normal situation, server loads and used network bandwidth should not exceed 50 % of the total capacity. Use a load balancer with a built-in SYN flooding protection. Use clustering to improve the performance and availability of the servers.

- Deploy a SIP-aware firewall (e.g. Secure SIP) at the entry point to the network. Alternatively, a transparent traffic repeater such as VoIP Defender could be used in front of the proxies. Fine-tune the selected device to have an optimum fit to the target environment.
- Keep all the software components and especially the SIP proxy up-to-date by patching them regularly. Harden the servers by disabling unused services and use a vendor recommended secure configuration.
- Separate the proxy and registrar servers. Enforce a quota for the registrations.
- Authenticate all SIP requests using predictive nonces. Verify user identities securely when they register to use the service. Maintain a list of suspicious users at the proxy and block them as needed.
- Use a SIP parser which has been proven to be robust and fast. Also, the memory allocation scheme the proxy uses should be efficient. Implement a watchdog process to restart the SIP software in case it crashes.
- Automatically switch off the support for RFC 2543 messages when the proxy is under a heavy load. For an extreme situation, form a list of VIP users whose connections are the only ones allowed. Prepare a related list of the allowed source IP addresses for the ISP to use in a filtering rule in their routers.
- Setup a dedicated and specialized DNS cache to protect the proxy against DNS flooding. Use low timeout values when possible.
- Use a signature-based, SIP-aware IPS in front of the SIP proxy. Forward alerts and logs to the operator console to analyze possible attacks against the SIP network and adjust the defenses as needed.
- Prefer TCP as the transport protocol for the SIP signaling if enough UAs support it to combat IP spoofing over UDP. For example, a switch to “TCP only” mode could be made under a heavy load using a dynamic DNS update, if the SRV records had low Time-to-Live (TTL) values.
- Finally, test thoroughly and regularly the deployed protection mechanisms.

In essence, an effective defense strategy includes using a combination of different defense methods, i.e., a layered defense model. Also, it is worthwhile to realize that the defense methods should be dimensioned based on the perceived risks since there are performance overhead and costs associated with the mechanisms.

5.5 Summary

This chapter presented different methods for protecting against the DoS attacks. First, application independent defenses were discussed. Next, the SIP-specific protection possibilities were investigated. Once introduced, all the countermeasures described in this chapter were rated for their effectiveness and implementation effort and placed in the comparison matrix.

Later on, based on the comparison matrix and the author's own personal experience, recommendations were given for the defense methods maximizing the protection against the attacks while keeping the costs low. It was found out that no single mechanism is sufficient alone, but instead a layered defense is preferred. Further, it takes time to adapt the selected protection methods to the prevailing conditions under which the service provider operates.

As many protection schemes would require almost universal cooperation among ISPs and ASs or protocol changes, these schemes are very difficult to realize. Therefore, these “external” methods being outside the control of a service provider, the service provider is generally limited to using merely local defenses against the attacks. Besides, DDoS attacks with armies of zombie hosts using non-spoofed addresses still remain an unsolved problem.

6 Conclusion

The object of this Master's thesis was to evaluate the chances of providing real-time SIP-based services in a potentially hostile Internet environment. Compared to a closed PSTN environment, malicious activities are easier to conduct in an open Internet environment where end points can communicate freely with (and possibly attack) other end points. In addition, it is relatively easy to forge one's identity over the Internet.

Denial of Service (DoS) attacks are a major threat to the users in the Internet, aiming to disrupt the services offered. Further, the DoS attacks are very hard to defend against as compared to many other threats (e.g. eavesdropping) in the Internet. Fortunately, as many of the attacks are similar to those facing the other protocols operating in the Internet, the defenses are very alike. These defense mechanisms were discussed and evaluated in detail from the viewpoint of a service provider. The solutions were compared according to their feasibility of implementation and effectiveness against the DoS attacks in the SIP service context.

Although the SIP protocol provides plenty of possibilities for the attacks, it is feasible to defend the SIP services against most types of DoS attacks. The results from this evaluation show that a layered defense model is a recommended defense strategy. Since the message flooding was found to be the most prominent threat and hardest attack type to handle, every SIP network should have a protection against flooding. Also, the use of authentication is essential together with a robust SIP parser. Finally, the defenses should be tested and adapted to the prevailing conditions under which the service provider operates.

Despite the fact that the author found no evidence of major DoS attacks against SIP services, it is worth noticing that there is one attack type for which no proper solution currently exists. Namely, the DDoS attacks using thousands of zombie hosts remain an unsolved threat. Ideally the flooding attacks should be prevented as close to their sources as possible, but due to the lack of cooperation among ASs and ISPs, the victim is in practice limited to using local defenses against these attacks. This problem could be diminished if, for instance, the ISPs would educate their customers in the information security matters so that their computers would not end up in the bot networks.

In the author's opinion this thesis has fulfilled the goals set in the beginning, but a limiting factor has been the lack of knowledge of the SIP landscape: the popular service providers, proxy implementations, common practices in the cross-domain communications, etc. More in-depth information about the SIP usage would have helped in drafting the recommendations.

If further studies on this subject are conducted, the recommended defense mechanisms could be validated in practice with simulated DoS attacks. Also, an up-to-date analysis of the general VoIP service availability and the DoS activity against SIP services in the Internet would undoubtedly yield intriguing results.

Future work could also include comparing Secure SIP and VoIP Defender in the same testbed subjected to the same attacks. The results would help to decide which one should be selected as the base architecture when protecting the SIP proxy.

References

Andersen, D. et al. (2001). "Resilient overlay networks", *Proceedings of the 18th ACM Symposium on Operating Systems Principles. SOSP '01*, p. 131.

Arkko, J. et al. (2003). Security Mechanism Agreement for the Session Initiation Protocol (SIP). IETF RFC 3329.

Baugher, M. et al. (2004). *The Secure Real-time Transport Protocol (SRTP)*. IETF RFC 3711.

Berners-Lee, T. et al. (1998). *Uniform Resource Identifiers (URI): Generic Syntax.* IETF RFC 2396.

Borenstein, N. and Freed, N. (1996). *Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies.* IETF RFC 2045.

CERT. (1998). *CERT Advisory CA-1998-01 Smurf IP Denial-of-Service Attacks*. Retrieved May 22, 2009 from http://www.cert.org/advisories/CA-1998-01.html

Cha, H. et al. (2008). "Detection of SIP De-Registration and Call-Disruption Attacks Using a Retransmission Mechanism and a Countermeasure Scheme", *IEEE International Conference on Signal Image Technology and Internet Based Systems, 2008. SITIS '08*, p. 650.

Dahlin, M. et al. (2003). “End-to-end WAN service availability”, *IEEE/ACM Transactions on Networking*, vol. 11, no. 2, pp. 300-313.

Eddy, W.M. (2006). "Defenses Against TCP SYN Flooding Attacks", *The Internet Protocol Journal*, vol. 9, no. 4, pp. 2-16.

Ehlert, S. et al. (2008a). "Two layer Denial of Service prevention on SIP VoIP infrastructures", *Computer Communications*, vol. 31, no. 10, pp. 2443-2456.

Ehlert, S. et al. (2008b). "Specification-Based Denial-of-Service Detection for SIP Voice-over-IP Networks", *The Third International Conference on Internet Monitoring and Protection, 2008. ICIMP '08*, p. 59.

Fiedler, J. et al. (2007). "VoIP Defender: Highly Scalable SIP-based Security Architecture", *Proceedings of the 1st international Conference on Principles, Systems and Applications of IP Telecommunications. IPTComm '07*, p. 11.

Fielding, R. et al. (1999). *Hypertext Transfer Protocol – HTTP/1.1.* IETF RFC 2616.

Franks, J. et al. (1999). *HTTP Authentication: Basic and Digest Access Authentication*. IETF RFC 2617.

Geneiatakis, D. and Lambrinoudakis, C. (2007). "A lightweight protection mechanism against signaling attacks in a SIP-based VoIP environment", *Telecommunication Systems*, vol. 36, no. 4, pp. 153-159.

Handley, M. et al. (1999). *SIP: Session Initiation Protocol*. IETF RFC 2543 (obsolete).

Handley, M. and Greenhalgh, A. (2004). "Steps Towards a DoS-resistant Internet Architecture", *Proceedings of the ACM SIGCOMM Workshop on Future Directions in Network Architecture. FDNA '04*, p. 49.

Handley, M. et al. (2006). *SDP: Session Description Protocol*. IETF RFC 4566.

Handley, M. and Rescorla, E. (2006). *Internet Denial-of-Service Considerations*. IETF RFC 4732.

Hersent, O. et al. (2000). *IP Telephony.* Addison-Wesley.

Jiang, W. and Schulzrinne, H. (2003). "Assessment of VoIP Service Availability in the Current Internet", *Passive and Active Measurement Workshop, PAM2003*.

Jung, J. et al. (2002). "DNS performance and the effectiveness of caching", *IEEE/ACM Transactions on Networking*, vol. 10, no. 5, pp. 589-603.

Kent, S. and Atkinson, R. (1998). *Security Architecture for the Internet Protocol.* IETF RFC 2401.

Kim, J. et al. (2008). "Implementation and Evaluation of SIP-Based Secure VoIP Communication System", *IEEE/IFIP International Conference on Embedded and Ubiquitous Computing. EUC '08*, p. 356.

Lau, F. et al. (2000). "Distributed Denial of Service Attacks", *IEEE International Conference on Systems, Man, and Cybernetics*, p. 2275.

Lipson, H.F. (2002). "Tracking and Tracing Cyber-Attacks: Technical Challenges and Global Policy Issues", *Special Report CMU/SEI-2002-SR-009,* Software Engineering Institute, Carnegie Mellon University.

Message Sequence Chart (MSC). Retrieved May 2, 2009 from http://www.sdl-forum.org/MSC/index.htm

Moore, D. et al. (2006). "Inferring Internet denial-of-service activity", *ACM Transactions on Computer System*s, vol. 24, no. 2, pp. 115-139.

Oftel. (2002). *Mobile networks call success rate surveys*. Retrieved May 15, 2009 from http://www.ofcom.org.uk/static/archive/oftel/publications/research/2002/call_survey/index1202.htm

Ormazabal, G. et al. (2008). "Secure SIP: A Scalable Prevention Mechanism for DoS Attacks on SIP Based VoIP Systems", *Proceedings of the 2nd international Conference on Principles, Systems and Applications of IP Telecommunications. IPTComm '08*, p. 107.

Peer-to-Peer Session Initiation Protocol (P2PSIP). Retrieved May 2, 2009 from http://www.ietf.org/html.charters/p2psip-charter.html

Peng, T. et al. (2007). "Survey of network-based defense mechanisms countering the DoS and DDoS problems", *ACM Computing Surveys*, vol. 39, no. 1.

Peterson, J. and Jennings, C. (2006). *Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP)*. IETF RFC 4474.

Poulsen, K. (2004). *FBI busts alleged DDoS Mafia*. Retrieved May 26, 2009 from http://www.securityfocus.com/news/9411

PROTOS - Security Testing of Protocol Implementations. Retrieved October 16, 2009 from http://www.ee.oulu.fi/research/ouspg/protos

Rescorla, E. and Modadugu, N. (2006). *Datagram Transport Layer Security*. IETF RFC 4347.

Rosenberg, J. (2001). *Request Header Integrity in SIP and HTTP Digest using Predictive Nonces*. IETF Internet Draft, draft-rosenberg-sip-http-pnonce-00 (work in progress).

Rosenberg, J. et al. (2002). *SIP: Session Initiation Protocol*. IETF RFC 3261.

Rosenberg, J. and Schulzrinne, H. (2002). *Session Initiation Protocol (SIP): Locating SIP Servers*. IETF RFC 3263.

Salsano, S. et al. (2002). “SIP Security Issues: The SIP Authentication Procedure and its Processing Load”, *IEEE Network*, vol. 16, no. 6, pp. 38-44.

Schulzrinne, H. et al. (2003). *RTP: A Transport Protocol for Real-Time Applications.* IETF RFC 3550.

Schulzrinne, H. (2008). *Session Initiation Protocol (SIP)*. SIP web site at Columbia University. Retrieved April 23, 2009 from http://www.cs.columbia.edu/sip/

Sicker, D.C. and Lookabaugh, T. (2004). “VoIP Security: Not an Afterthought”, *Queue*, vol. 2, no. 6, p. 56.

Sisalem, D. et al. (2006). "Denial of service attacks targeting a SIP VoIP infrastructure: attack scenarios and prevention mechanisms", *IEEE Network*, vol. 20, no. 5, pp. 26-31.

Sparks, R. (2007). “SIP: Basics and Beyond”, *Queue*, vol. 5, no. 2, pp. 22-33.

Stevens, R. (1994). *TCP/IP Illustrated, Volume 1: The Protocols.* Addison-Wesley.

Third Generation Partnership Project (3GPP), http://www.3gpp.org

Tschofenig, H. and Rescorla, E. (2006). *Real-Time Transport Protocol (RTP) over Datagram Transport Layer Security (DTLS)*. IETF Internet Draft, draft-tschofenig-avt-rtp-dtls-00 (work in progress).

Vaughn, R. and Evron, G. (2006). *DNS Amplification Attacks.* Retrieved May 25, 2009 from http://www.isotf.org/news/DNS-Amplification-Attacks.pdf

Wieser, C. et al. (2003). “Security testing of SIP implementations”, *Technical Report CUCS-024-03*, Department of Computer Science, Columbia University.

Zhang, G. et al. (2007). "Denial of service attack and prevention on SIP VoIP infrastructures using DNS flooding", *Proceedings of the 1st international Conference on Principles, Systems and Applications of IP Telecommunications. IPTComm '07*, p. 57.

Appendix A: SIP Response Codes

PROVISIONAL	REDIRECTION
"100" Trying	"300" Multiple Choices
"180" Ringing	"301" Moved Permanently
"181" Call Is Being Forwarded	"302" Moved Temporarily
"182" Queued	"305" Use Proxy
"183" Session Progress	"380" Alternative Service
SUCCESS	
"200" OK	
CLIENT ERROR	
"400" Bad Request	"420" Bad Extension
"401" Unauthorized	"421" Extension Required
"402" Payment Required	"423" Interval Too Brief
"403" Forbidden	"480" Temporarily Unavailable
"404" Not Found	"481" Call/Transaction Does Not Exist
"405" Method Not Allowed	"482" Loop Detected
"406" Not Acceptable	"483" Too Many Hops
"407" Proxy Authentication Required	"484" Address Incomplete
"408" Request Timeout	"485" Ambiguous
"410" Gone	"486" Busy Here
"413" Request Entity Too Large	"487" Request Terminated
"414" Request-URI Too Long	"488" Not Acceptable Here
"415" Unsupported Media Type	"491" Request Pending
"416" Unsupported URI Scheme	"493" Undecipherable
SERVER ERROR	**GLOBAL FAILURE**
"500" Server Internal Error	"600" Busy Everywhere
"501" Not Implemented	"603" Decline
"502" Bad Gateway	"604" Does Not Exist Anywhere
"503" Service Unavailable	"606" Not Acceptable
"504" Server Time-out	
"505" Version Not Supported	
"513" Message Too Large	