Security of VoIP

Analysis, Testing and Mitigation of SIP-based DDoS attacks on VoIP Networks

A thesis submitted in partial fulfilment of the
requirements for the Degree
of Master of Science in Computer Science
in the University of Canterbury

by Xianglin Deng
University of Canterbury

2008

Abstract

Voice over IP (VoIP) is gaining more popularity in today's communications. The Session Initiation Protocol (SIP) is becoming one of the dominant VoIP signalling protocol[1, 2], however it is vulnerable to many kinds of attacks. Among these attacks, flood-based denial of service attacks have been identified as the major threat to SIP. Even though a great deal of research has been carried out to mitigate denial of service attacks, only a small proportion has been specific to SIP. This project examines the way denial of service attacks affect the performance of a SIP-based system and two evolutionary solutions to this problem that build on each other are proposed with experimental results to demonstrate the effectiveness of each solution.

In stage one, this project proposes the Security-Enhanced SIP System (SESS), which contains a security-enhanced firewall, which evolved from the work of stage one and a security-enhanced SIP proxy server. This approach helps to improve the Quality-of-Service (QoS) of legitimate users during the SIP flooding attack, while maintaining a 100 percent success rate in blocking attack traffic. However, this system only mitigates SIP INVITE and REGISTER floods.

In stage two, this project further advances SESS, and proposes an Improved Security-Enhanced SIP System (ISESS). ISESS advances the solution by blocking other SIP request floods, for example CANCEL, OK and BYE flood.

JAIN Service Logic Execution Environment (JAIN SLEE) is a java-based application server specifically designed for event-driven applications. JAIN SLEE is used to implement enhancements of the SIP proxy server, as it is becoming a popular choice in implementing communication applications.

The experimental results show that during a SIP flood, ISESS cannot only drop all attack packets but also the call setup delay of legitimate users can be improved substantially compared to and unsecured VoIP system.

Acknowledgement

I would like to thank my supervisor, Associate Professor Ray Hunt for supervising my project and Dr. Malcolm Shore of Telecom who has given me a lot of encouragement and has supported me throughout my thesis with his patience and knowledge.

Many thanks go to my colleagues at the lab for their generous help, and sharing ideas with me. Furthermore, I would like to thank all my friends in the Computer Science department.

I would like to thank Chris Chou, for believing in me, and cheering me up in difficult times.

Finally, I thank my mother for supporting me throughout all my studies at University, and being the consultant of my life.

Contents

Chapter 1: Introduction

Voice over IP (VoIP) is an increasingly popular form of voice communication. In VoIP call setup and management operations are completed through signalling messages and most modern VoIP systems use the Session Initiation Protocol (SIP) [3] for the signalling process [4]. However, SIP is vulnerable to many kinds of attacks [5] [6] [7] [8] [9] [10] among which flood-based Denial of Service (DoS) attack [11] is identified as the biggest threat [9] [12] [13]. For example, asterisk (an open source SIP-based VoIP switch) is used by some organizations to establish VoIP calls between internal users and external users. Since the transmission link between the internal and external users is the internet, the VoIP switch is vulnerable to attacks sfrom the internet. Even though a great deal of research [14] [15] [16] [17] [18] [19] has been carried out to mitigate DoS attacks, only a fraction of this work is specific to SIP, further more, the existing solutions have their limitations in terms of complexity, accuracy and so on. SIP flood protection is only handled in a very limited manner by the majority of firewalls, thus there is much work remaining to be done.

1.1 Objective and Approach

There are two major types of SIP-based VoIP deployment: VoIP on a purely private network, and VoIP on an open Internet. When VoIP is deployed on a purely private network, it is normally highly integrated with a PSTN network and VPN, where users cannot access the system from outside. This can protect the system from external attacks, however it cannot stop attacks from the internal network. If this system is deployed as a public service, and can be accessed via the

Internet, it is susceptible to flood attacks from both internal and external users. Even though the topological implementations are different on the different types of SIP-based VoIP deployment, the attack mechanism and impact are similar in both systems.

The objective of this thesis is to find a solution to mitigate SIP flooding attacks, which is able to drop the majority of attack packets while continuing to provide a good QoS for legitimate users. The approach used in this project is to develop a protocol and verify its performance using a VoIP testbed.

This project firstly examines the impact of a SIP flooding attack on a SIP-based VoIP system. In this system, a SIP proxy server is in charge of forwarding SIP requests and responses to the corresponding recipients, and is most vulnerable to flooding attacks, because it has to process each incoming SIP request, look up the address of the recipient and it may need to generate, store and send authentication requests. While there are a number of types of SIP requests that can be used to flood the SIP proxy server, in this project we focus on the INVITE flood as an example to illustrate the impact of this attack, because INVITE and REGISTER requests require more processing compared to other SIP requests, and the behaviour of INVITE and REGISTER requests are very similar. For simplicity, we will mainly use INVITE requests to illustrate the impact of SIP floods. Our objective, however, is to deliver a protocol which addresses a wide range of SIP flooding attacks, such as the INVITE flood, the CANCEL flood and the OK flood.

Having demonstrated and established the impact of the SIP INVITE flooding attack, we further examined a couple of commercial firewalls' performances against SIP flood attacks. Experimental results showed that the SIP flood attacks can defeat the security mechanisms

of the tested firewalls. The implication is that any businesses using these firewalls are vulnerable to SIP flood attacks. Furthermore, any other commercial firewalls with similar security mechanisms are also vulnerable to this type of attack. In order to mitigate SIP flood attacks, two evolutionary solutions are proposed, and each solution's advantages and drawbacks are discussed and verified with experimental results.

In stage one, this project proposes a Security-Enhanced SIP System (SESS), which contains a security-enhanced firewall evolved from an application-layer stateless firewall with additional layer-3 queuing and a security-enhanced SIP proxy server. This approach is an advance on the previous one in that it improves the QoS of legitimate users during a flooding attack.

In stage two, this project further evolves SESS, and proposes an Improved Security-Enhanced SIP System (ISESS). ISESS advances the solution by blocking other SIP request floods, for example CANCEL, OK and BYE floods.

JAIN SLEE is used to implement enhancement of the SIP proxy server. This is because JAIN SLEE provides high performance and low latency for communication applications. Additionally it uses Java, a high level language, which reduces the implementation time.

. Experimental results verify that the final solution is able to block all types of spoofed SIP requests, while maintaining a good QoS for legitimate users.

1.2 Thesis Structure

This thesis is structured as follows:

Chapter two provides an overview of VoIP, and the general information of SIP, followed by the common threats to a SIP-based VoIP system. It is important to note that this project focuses on flood-

based SIP DoS attacks, thus a few existing SIP flooding attack mitigation techniques will be described and analyzed. Their advantages and drawbacks will be discussed.

Chapter three discusses SIP flooding attacks in detail, followed by descriptions of existing mitigation techniques. The performance of a couple of commercial firewalls is examined using a VoIP testbed, and the vulnerabilities of these firewalls are identified.

Chapter four details a proposed solution- the Security-Enhanced SIP system (SESS) which mitigates spoofed SIP INVITE and REGISTER flood while maintaining a good Quality-of-Service for legitimate users. The advantages and drawbacks of SESS are discussed. Prior to SESS, an application-layer stateless firewall is proposed to stop spoofed INVITE and REGISTER floods, which SESS is based on. The details of the application-layer stateless firewall are discussed and its performance is examined using our VoIP testbed.

Chapter five provides an explanation for an improved solution based on SESS – Improved Security-Enhanced SIP system (ISESS). ISESS is an advance on SESS in that it eliminates its drawbacks, while still maintaining the advantages of SESS.

Chapter six describes the implementation process of SESS and ISESS, followed by a series of experiments. Experimental results are carefully analysed. Experimental results show that by using ISESS the objectives of this project can be achieved.

Chapter seven is the conclusion section and suggestions for future work are also discussed.

Chapter 2

Background

2.1 VoIP overview

The term Voice over IP (VoIP) is used to describe the technology for enabling voice communication over IP networks to a similar level of functionality and quality as is available on a traditional public switched telephony network (PSTN). VoIP technology employs a suite of protocols which can be categorized into signalling and data transfer protocols. There is a strong business and consumer interest in VoIP owing to its potential for providing a more flexible service at a much lower cost than is typically available from analogue telephony. However, as it is built on standard IP networks, it is vulnerable to the wide range of network attacks associated with the Internet, such as DoS, eavesdropping, virus infection, trojans etc [10].

This thesis focuses specifically on SIP-based flooding which is one of the more common ways to attack SIP systems.

2.1.1 Quality of Service (QoS) and security requirements of VoIP

VoIP faces two challenges which are more serious than in traditional PSTN networks: quality of service and security. Owing to the fact that in VoIP networks there is typically a great deal of infrastructure resource sharing, the quality of a VoIP network cannot be guaranteed to the same extent as in the PSTN network. Service quality on a VoIP network consists of the following factors [20]: Network Availability, Latency, Jitter and Packet Loss. In this project, call setup delays will be measured as the main system performance

factor. Call setup delay indicates the time takes to setup a phone call. This can reflect the latency of the network during the call setup phase, and the call setup timeout rates can indicate the network availability. Jitter and packet loss rate are not measured due to the limitation of the experimental measuring tool.

2.1.2 VoIP protocol stack

VoIP protocols can be divided into two categories: Signalling and voice transmission protocols. Figure 1 [21] shows the essential protocols in a typical VoIP protocol stack.

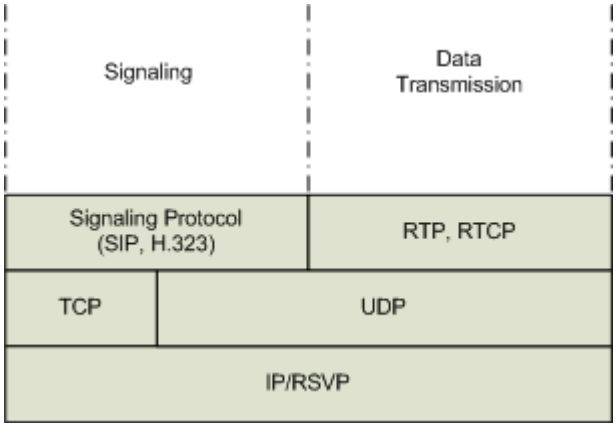


Figure 1: Essential protocols in a VoIP protocol stack

The signalling protocols are in charge of setting up, managing, controlling and terminating a session. The voice transmission protocols are responsible for transmitting the actual voice data across the network. In the following section, we will discuss the main VoIP signalling protocols (H.323 [22] and Session Initiation Protocol (SIP) [23]) in detail. The vulnerabilities of VoIP will also be described.

2.1.2.1 Signalling protocols

Both H.323 and SIP provide functionalities for call setup, management, and termination. These protocols enable amongst other

things negotiation of the codec to be used in voice data encoding and the delivery mechanisms (e.g. RTP [24] over UDP/IP [25]) for both protocols. The following subsections detail the call setup and management in the two protocols,

2.1.2.1.1 H.323

H.323 is a protocol suite that was designed to enable IP-based multimedia communications, and it was the first widely adopted and deployed VoIP protocol. Figure 2 [22] shows a H.323 protocol suite.

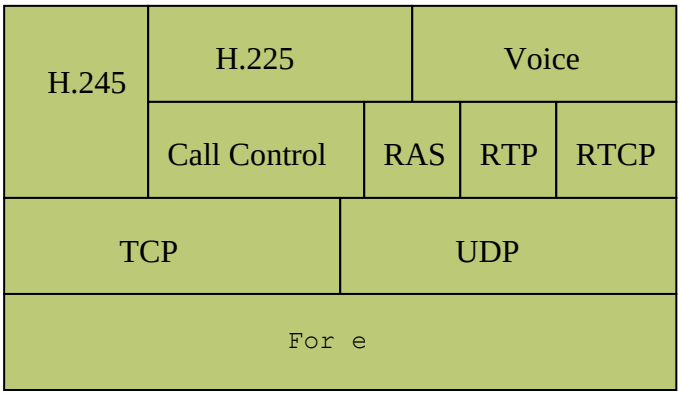


Figure 2: H.323 protocol suite

The core protocols contained in H.323 suite are:

- **H.245** [26] for opening and closing logic channels for each multimedia session; H.245 is also in charge of capacity and codec negotiation. Two H.323 end points can set up a fast connection without a gatekeeper, by exchanging H.245 messages.
- **H.225** [27] for call setup, alert, connecting, and call termination;
- **RAS** [22] (Registration, Admission, Status) is used to phone management. RAS establishes logical channels between phones and gatekeepers that manage these phones. Without appropriate RAS communication, a phone cannot place or receive phone calls.
- **RTP** is used for sending or receiving multimedia information.

Drawbacks of H.323

While H.323 is the most widely used VoIP protocol suite, it has a number of drawbacks. The major one is the lack of scalability: H.323 was originally designed to be used on a LAN. The newest version of H.323 defines methods for locating users across a zone. However, when there are multiple domains, H.323 has a scalability problem as there is no easy way to perform loop detection. Another drawback to using H.323 is complexity which stems from the use of several protocol components. This also complicates firewall traversal, as firewalls must act as application level proxies [28], parsing the entire message to arrive at the required fields. Furthermore, H.323 has poor extensibility, which means it is hard to develop additional extensions for this protocol.

Since this project will only focus on SIP-based VoIP systems, the details of H.323 will not be discussed in detail in this document.

2.1.2.1.2 Session Initiation Protocol

SIP is a lightweight application layer protocol designed to manage and establish multimedia sessions such as video conferencing, voice calls, and data sharing through requests and responses. It is increasingly gaining favour over H.323 in the VoIP environment. Three advantages of SIP are:

- It uses Uniform Resource Locators (URL) [29] addressing scheme, which is physical location independent. Addressing can be a phone number, an IP address, or an e-mail address. The messages are very similar to those used by the Internet (HTTP [30]).
- It allows multiple media sessions during one call. This means that users can share a game, instant message (IM), and talk at the same time.
- It is a “light” protocol and is easily scaleable.

Packetizer [31] and Schulzrinne et al. [4] have compared the performance of these two protocols thoroughly. Schulzrinne concludes that even though H.323 and SIP provide similar functionality, SIP is a better candidate for VoIP in terms of simplicity, extensibility and scalability.

Since SIP is just an application layer signalling protocol, many security mechanisms are optional and little attention has been given to SIP security features [9].

The following section describes the SIP-based VoIP systems in detail.

2.2 SIP-based VoIP systems

This section will firstly provide an overview of SIP messages and SIP components, and then explain the SIP processes in detail. A detailed review of the threats to and security of the SIP protocol is studied.

2.2.1 SIP components

A SIP-based VoIP system contains the following four essential components:

- User Agent (UA) is the component interacting with the end user to complete a SIP request. A SIP client can act as both a SIP user agent client (UAC) and a SIP user agent server (UAS), where the UAC generates outgoing SIP requests, and UAS handles incoming SIP requests.
- SIP proxy server: the SIP proxy server receives SIP requests from various user agents and forwards them to the appropriate hosts. It may also contain an authentication function;
- Registrar server: It processes REGISTER messages (described in the next section), and it maps the users URI to their current location. For example, 2001@testbed.com may be mapped to 2001@192.168.2.4:5060, where 192.168.2.4 is the current IP

address of the client 2001, and 5060 is the port on which his SIP UA is listening. In some systems, the registrar server is located on the SIP proxy server.

- Location Server: A location server is used to store the locations of registered users. It is used by a proxy to find the destination client's possible location. This function is most often performed by the registrar server.

There are also some other components in a SIP-based VoIP system, for example Redirect server; however we will not discuss them in this project as they are not essential to the VoIP system.

2.2.2 SIP Messages

SIP uses header messages similar to HTTP [30] to communicate. The message body is either used to describe session requirements or to encapsulate various types of signalling. SIP addresses follow the general form of email addresses; an example of a SIP address is sip:2001@testbed.com. The text-based presentation of a SIP message makes it more vulnerable to attacks. Figure 3 shows a typical SIP INVITE message, and Figure 4 shows a typical REGISTER message.

```
Request-Line: INVITE sip:2002@opencloud.com
      Method: INVITE
      [Recent Packet: False]
Message Header
      Via: SIP/2.0/UDP 10.0.0.34:5060; rport;
      branch=z9hG4bK56612D86EA77e51A
      Max-Forwards: 70
      From: 2003<sip:2003@testbed.com>;tag=301012803
      To:2002<sip:2002@testbed.com>
      Contact: <sip:2003@10.0.0.34:5060>
      Call-ID:-327e-jki398slmen@10.0.0.34
      CSeq:2 INVITE
      Content-Type: application/sdp
      User-Agent: Elite 1.0 Brcm callctrl/1.5.1.0
      Content-Length:458
      Supported: timer
      Allow: NOTIFY
      Allow: REFER
      Allow: OPTIONS
      Allow: INVITE
```

Figure 3: typical SIP INVITE message

```
Request-Line: REGISTER sip:2003@opencloud.com
      Method: REGISTER
      [Recent Packet: False]
Message Header
      Via: SIP/2.0/UDP 10.0.0.34:5060; rport;
      branch=z9hG4bK56612D86EA77e51A
      Max-Forwards: 70
      From: 2003<sip:2003@testbed.com>;tag=301012803
      To:2003<sip:2003@10.0.0.34:5060>
      Contact: <sip:2003@10.0.0.34:5060>
      Call-ID:so98-8834-327e-jki398slmen@10.0.0.34
      CSeq:1 REGISTER
      Expires: 3600
      Content-Length: 0
```

Figure 4: typical SIP REGISTER message

There are two types of SIP messages: request, and response to a corresponding request message. Request messages are used by UAC, and responses are used by UAS. When a userA wants to make a phone call to userB, userA's UAC will generate an INVITE message, and send it to userB's UAS (it may or may not be via a SIP proxy server),

Then, userB's UAS will process that request, and send corresponding responses. Table 1 shows the common SIP request messages.

Table1: Common SIP Requests

SIP Request	Purpose
INVITE	To initiate a session
BYE	To terminate an existing session
OPTIONS	To determine the SIP messages and codecs that the UA or server understands
REGISTER	To register a location from a SIP user
ACK	To acknowledge a response from an INVITE request
CANCEL	To cancel a pending INVITE request (it is important to note that this operation does not affect a completed request?)
SUBSCRIBE	To indicate the desire for future NOTIFY requests
NOTIFY	To provide information about a state change that is not related to a specific session. (For example, Windows instant messenger uses NOTIFY to transfer group information.)
REFER	To transfer calls and contact external resources

SIP responses are three-digit codes similar to HTTP (for example, 404 Not Found, and 200 OK). The first digit indicates the category of the responses. There are 6 categories, namely: information responses (1xx), successful responses (2xx), redirect responses (3xx), request failure (4xx), server failure (5xx) and global failure (6xx). There are dozens of response messages, and table 2 shows only a few very common SIP responses.

Table 2: Brief overview of SIP responses

Response	Purpose
100 Trying	To indicate a proxy has received an INVITE request, and is processing it.
180 Ringing	The INVITE has been forwarded to the destination
200 OK	A session has been set up
401 Unauthorized	A response to a REGISTER request, if the user did not provide correct authentication information
407 Proxy Authentication Required	A response to an INVITE request, if authentication is enabled on the proxy, and the user did not provide correct authentication information
408 Request timeout	To indicate there is no response to a request within a certain time
503 Service unavailable	To indicate the current request cannot be processed

2.2.3 SIP process

To explain how SIP components interact with each other using SIP messages, this section will discuss the processes involved in a SIP-based VoIP system. Figure 5 shows the flow of interaction of a SIP-based VoIP system.

The main SIP operations involved in a VoIP system are:

- *Registration*: If a user agent wants to receive phone calls, he has to register with the registrar by sending a REGISTER request (Step 5 in figure 6).
- *Invite*: When a user wants to place a phone call, it will send an INVITE request to his proxy server (Step 2 in figure 5).

- The proxy server will process the request (the process will be explained in a later subsection), and forward it to the callee.
- When the callee picks up the phone, an OK response will be sent back to the caller. Then the session is set up.

It is important to note that, for simplicity, some minor message exchanges are not shown.

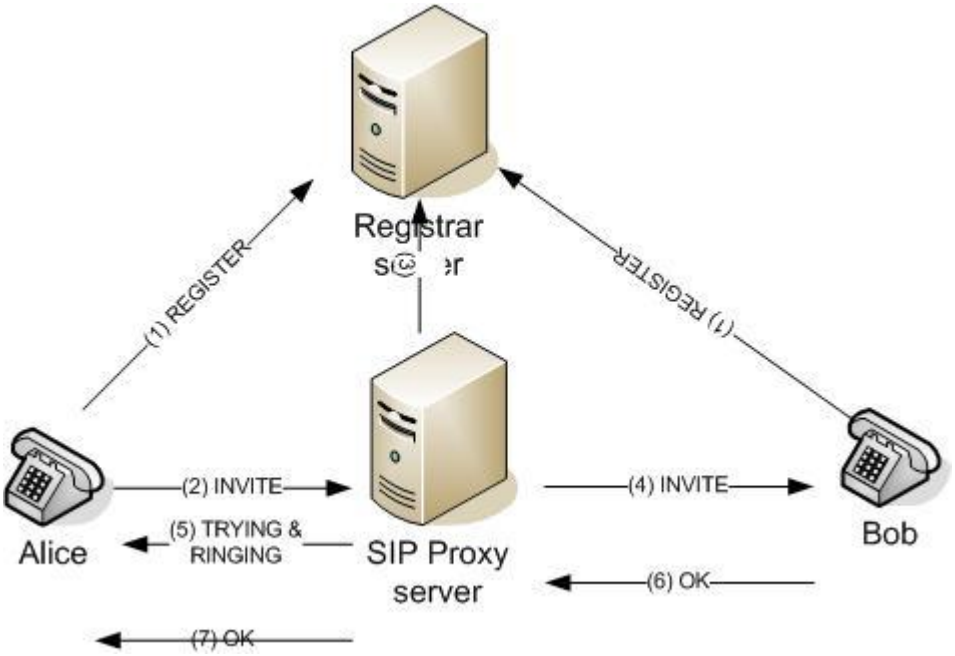


Figure 5: SIP operations

2.2.3.1 SIP proxy operations on INVITE request

A SIP proxy can operate in two models: authentication enabled, and no authentication. In order to receive phone calls from previously unknown callers (possibly globally), a SIP proxy has to disable authentication on INVITE requests. There is a unique field in the SIP header called CallID, which is a UAC generated random ID to identify a session. All subsequent requests and responses within that session

will carry the same CallID. When no authentication is required, the proxy does the following with an INVITE request:

- When an INVITE is received at proxy, the proxy will send a query to the location server, to find the actual contact address of the destination,
- When the INVITE is forwarded to the destination, 100 TRYING and 180 RINGING responses are sent back to the caller.
- As soon as the callee picks up the phone, an OK response is sent back to the caller.
- Finally, an acknowledgment (ACK) request is sent to the callee, then the voice session starts.

This process is slightly different if authentication is enabled on the proxy server. The behaviour of the authentication enabled proxy server will be discussed in the following subsection.

2.2.4 SIP authentication

As specified in RFC3261, SIP provides a challenge-response-based authentication using HTTP digest authentication. Using this mechanism the SIP user agent client (UAC) is able to identify itself to a user agent server (UAS) (or proxy server or registrar server). Therefore, SIP authentication applies only to user-to-user or user-to-proxy communications;

After the SIP proxy server receives the INVITE, instead of processing the INVITE request, the proxy server will send a 407 Authentication required response to challenge the caller. In the 407 message, there is a “nonce” value, which is a random string generated by the proxy server used for one challenge only. Both the SIP server (proxy, registrar) and UAC share a secret password, which is sometimes the password for the user. The caller uses the nonce, username, password and realm to create a unique response value. The

UAC sends the request again, including the computed response value, which is used by the server to authenticate the request. Using this mechanism, the password is never sent in clear text. An illustration of the digest authentication procedure is given in figure 6. MD5 is the default function used for computing the response by combining the input parameters [32]. This mechanism puts more processing load on the SIP proxy server, thus making it more vulnerable to flooding attacks.

In the following section, SIP-based VoIP system vulnerabilities will be discussed. Since this project focuses on SIP flood DoS attacks, chapter three will explain this type of attack in detail.

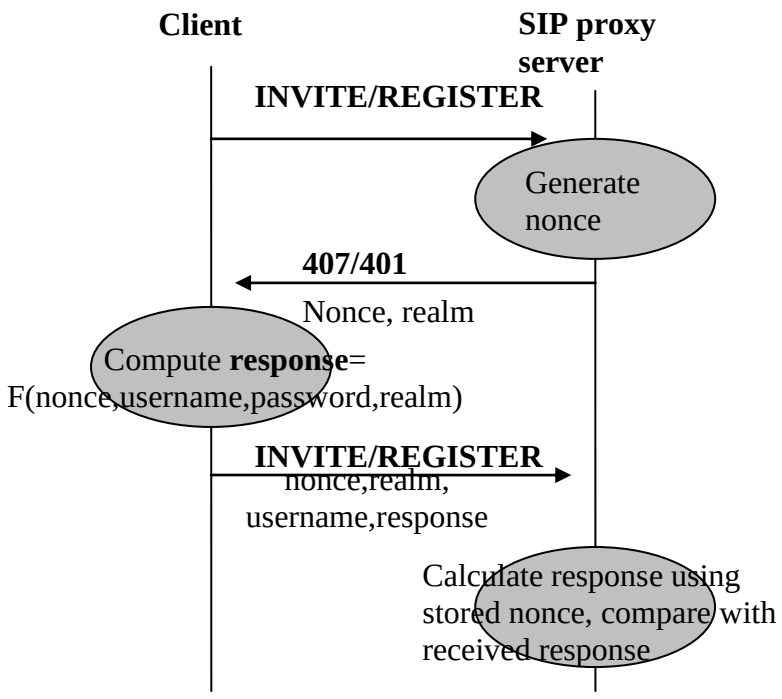


Figure 6: SIP proxy authentication process

2.3 SIP vulnerabilities

Like any other IP-based system, VoIP systems are susceptible to a variety of attacks [33, 34]. The most common VoIP attacks are: Eavesdropping [35], Flooding based denial of service attack [36] (The most common DoS attacks are: UDP flooding [37] and TCP SYN flooding [38] [39]), Packet fragmentation attack [40] [41](for example the ping of death [42]), RTP insertion attack [43], Fuzzing/Malformed message DoS attack [44] (which can be used to find a flaw in the target system and cause DoS on a VoIP entity), Spam over internet telephony (SIPT) [45, 46] (Even though this kind of attack is still very rare, a number of researchers have published work in this area [46-49]), and Voice Phishing (Vishing) [50] (Vishing is typically used in identity theft schemes such as cleverly impersonating highly trusted entities (banks), to obtain the personal and financial information of other users).

Since this project focuses on the SIP-based VoIP system, this thesis will discuss attacks specific to SIP in detail.

There are many lists of security threats that are specific to SIP-based VoIP systems [5] [6] [7] [8] [9] [10] [51] [52]. Salsano et al [9] identify that a SIP-based VoIP system is especially prone to DoS attacks. Based on a VoIP threat taxonomy compiled by VOIPSA [53], a DoS attack can be categorised into the following groups:

- Network bandwidth attack. A network bandwidth DoS attack simply floods a target with a large number of random packets in an attempt to congest its network bandwidth.
- OS/firmware attack. An OS/firmware DoS attack attempts to crash a target by exploiting some specific underlying OS/firmware vulnerabilities. It can also exhaust the target by over consuming OS/firmware resources, such as CPU and memory.

- SIP-function specific attack. This is an attack specific to some functions of SIP, such as call setup time.

Since work has already been carried out on the first two categories of attacks [54] [55] [15] [54] [56] [57] [16], this project will focus on the attacks that are specific to SIP.

In this section, we will list a few of the most common SIP application-layer security threats, and will explain the SIP flooding attacks in detail.

2.3.1 Signalling manipulation

There are several attacks in which an attacker manipulates a SIP signalling message to hijack or manipulate calls.

2.3.1.1 Registration removal

Registration removal can be done by modifying the REGISTER request [58]. There are two important fields in a REGISTER header, one is *Contact*, and the other is *Expires*. The contact header specifies the actual address that the registrant is listening on for incoming calls. Expires specifies when this registration expires. To remove a registration, the attacker needs to send a REGISTER message with **Contact** set to *****, and **Expires** set to **0**. Figure 7 shows a spoofed registration removal message.

2.3.1.2 Registration addition

The SIP registrar allows multiple contact addresses for one user, all of which can ring when an inbound call arrives. When multiple SIP phones ring, the first one to go off hook will answer the call. This behaviour creates the opportunity for several attacks. For example, an attacker can add multiple addresses to every registration and when

some one makes a phone call to one of them, multiple phones would ring, and this would cause chaos in an enterprise.

```
Request-Line: REGISTER sip:2003@opencloud.com
      Method: REGISTER
      [Recent Packet: False]
Message Header
      Via: SIP/2.0/UDP 10.0.0.34:5060; rport;
      branch=z9hG4bK56612D86EA77e51A
      Max-Forwards: 70
      From: 2003<sip:2003@testbed.com>;tag=301012803
      To:2003<sip:2003@10.0.0.34:5060>
      Contact: *
      Call-ID:82s98909-327e-jki398slmen@10.0.0.34
      CSeq:1 REGISTER
```

Figure 7: Registration removal message

2.3.1.3 Registration hijacking

Registration hijacking occurs when an attacker impersonates a valid UA to a registrar and replaces the legitimate registration with its own address. In SIP, a User Agent (UA) must register itself with a SIP proxy/registrar (or IP PBX), which allows the proxy to direct inbound calls to the UA. When a UA registers itself it sends a REGISTER request which contains the *Contact:* header which indicates the IP address of the user's device. The registrar would take the *Contact* as the binding address of the requesting UA. An attacker can replace the legitimate *Contact* with its own IP address. The effect of this attack is that all the inbound calls will be directed to the attacker's UA. Furthermore, registration is normally performed using UDP, which is more susceptible to spoofed attack. According to RFC 3261, not all registrars require authentication for the requesting UA, even if it does, the authentication mechanism is very weak (username and password). Thus this type of attack can be a big threat to a SIP-based VoIP system.

2.3.1.4 Signalling manipulation countermeasure

One way to mitigate the above signalling manipulation attack is to enable authentication on the registrar. Since REGISTER messages are not exchanged frequently, so the overhead for authentication is minimal. Authentication requires that only legitimate users can register (for example, people from the enterprise) and that strong passwords are used. This project will focus on signalling manipulation attacks, since this attack can be eliminated by enabling authentication.

2.3.2 Malformed message and countermeasure

Other DoS attack opportunities are caused by implementation flaws of SIP systems. A large number of systems are found to be vulnerable to malformed SIP messages [59]. Such DoS attack does not have a generalised impact on VoIP systems, because it can only target specific implementations or products. These vulnerabilities are typically short lived and easily fixed through software patches.

2.3.3 Flood-based DoS attack

A flooding-based DoS [60] attack can be achieved by using massive volumes of useless traffic to occupy all the resources that would otherwise be used to service legitimate traffic. If the attack traffic comes from multiple sources, it is called a Distributed DoS (DDoS). This type of DoS attack is hard to prevent, as the targets can be attacked simply because they are connected to the public Internet. As mentioned earlier [12], flood-based DoS attack is the biggest threat VoIP is facing and the remainder of this project focuses on these attacks and their mitigation. In Chapter three, the details of this type of attack will be discussed, followed by existing mitigation techniques.

Chapter 3: SIP flood attacks and existing countermeasures

As mentioned in section 1.1, SIP flood attacks are the major threat to VoIP systems. This chapter will explain how such attacks can affect the performance of the system, using an INVITE flood as an example, with an experimental verification. Later, a few existing SIP flood mitigation techniques will be examined and their advantages and drawbacks will be discussed.

3.1 Overview of SIP message flooding attack

A SIP message flooding attack occurs when an attacker sends a large number of INVITE or REGISTER requests with spoofed source IP addresses [61]. It is worth pointing out that even though there are many other types of SIP requests, INVITE and REGISTER are the predominant messages used by SIP[3], and they require more processing at the SIP components than all the other requests. Thus, SIP-based VoIP systems are especially vulnerable to flooding attacks using these requests. Figure 8 shows the message flow to setup a VoIP session.

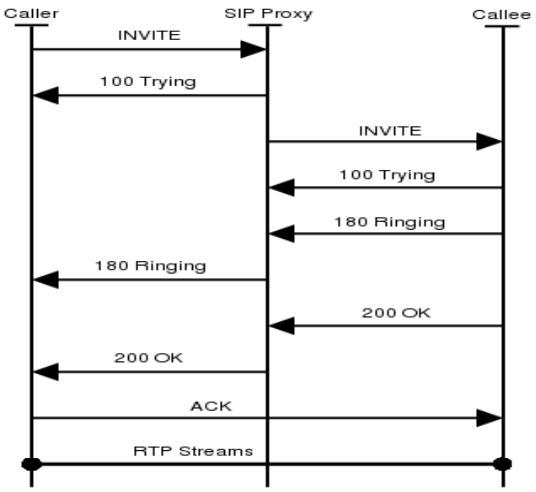


Figure 8: SIP call setup process

There are two major impacts resulting from a SIP flooding attack:

- **Memory exhaustion**: When a SIP proxy server receives a SIP request (REGISTER or INVITE) it needs to copy each incoming request into its internal buffers to be able to process the message. These messages will at least be kept till the last OK message is sent to terminate the call setup handshake. Also, the server normally keeps a copy of forwarded messages for further processing (for example, digest authentication). In some cases the server is configured as a stateful server, which will need to maintain information about the session throughout the lifetime of the session, for example when the communication path involves firewall or NAT traversal [62]. The size of SIP messages can vary from hundreds to thousands of bytes, and the call setup handshake normally lasts from 1 second to a few seconds if human interaction is required, which makes the proxy server vulnerable to memory exhaustion attacks.
- **CPU exhaustion**: After the incoming requests are saved, the SIP proxy server will process (authentication or destination address look-up etc.) the requests and generate and send responses. The

CPU resource can become highly loaded if a large number of requests are flooded at the SIP proxy server.

- **Link Bandwidth.** SIP flooding attacks can exhaust the link bandwidth of the SIP proxy server and cause a denial of service at the access point to the VoIP system.

While enabling authentication on the SIP proxy server will avoid some types of flooding attack, it requires more resources to process each incoming request (e.g. more RAM is needed to store generated nonce values and more CPU to calculate them). Hence any attack which can be mounted on an authenticating server will have a more devastating effect than would be the case on a non-authenticating server.

3.1.1 SIP Flooding Test Bed

In order to examine the effect of INVITE flooding attacks on the performance of a SIP proxy server, a VoIP test bed was established and an attack tool based on an INVITE Flooder [63], called iFlood [64] was developed.

The basic test bed is as shown in Figure 9:

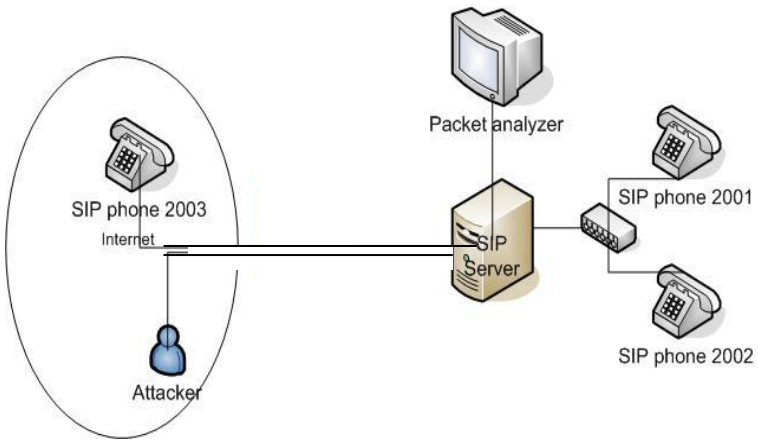


Figure 9: SIP Test bed Setup without Firewall

The initial SIP proxy server that was used was the Asterisk [1] public domain server. The SIP client software was an X-lite SIP soft phone.

iFlood is used to generate a large number of INVITE messages, with spoofed source IP addresses. The attack can specify the range of IP addresses to be spoofed, as well as the spoofed username. The iflood command to send INVITE flood is:

./iflood eth0 target_extension target_domain target_ip num_of_attack_packets spoofed_IP_range –S source_extension

eth0 is the network interface that the attack uses to send attack packets;
target_extension is the extension of the target host, it can either be a number, or a word, for example: 2002 or testbed02;
target_domain is the domain of the target host, in our testbed, the domain is testbed.com;
target_ip is the IP address of the target domain. However, it is worth noting that if a NAT-enabled firewall is used, the target_ip should be the IP address of the external interface of the firewall;
num_of_attack_packets is the total number of attack packets to be sent;
spoofed_IP_range is the range of IP addresses to be spoofed. There are two options: **random** or **ranged**. **Random** option means using randomly spoofed IP addresses. **Ranged** option allows the attacker to specify the range of IP addresses to be used. This option can be useful, if the firewall has ingress filter enabled; and
source_extension is the extension used by the spoofed SIP request.

[1] Asterisk® is the world's leading open source telephony engine and tool kit, and has the largest support community. For more information, please visit http://www.asterisk.org/

3.1.2 SIP Flood Test

The SIP proxy server was flooded with 60,000 INVITE packets at the attack machine's maximum rate of 3245 packets/second, and the call setup delays monitored during the attack. Figure 10 shows the call setup delay when the system is under SIP flooding attack.

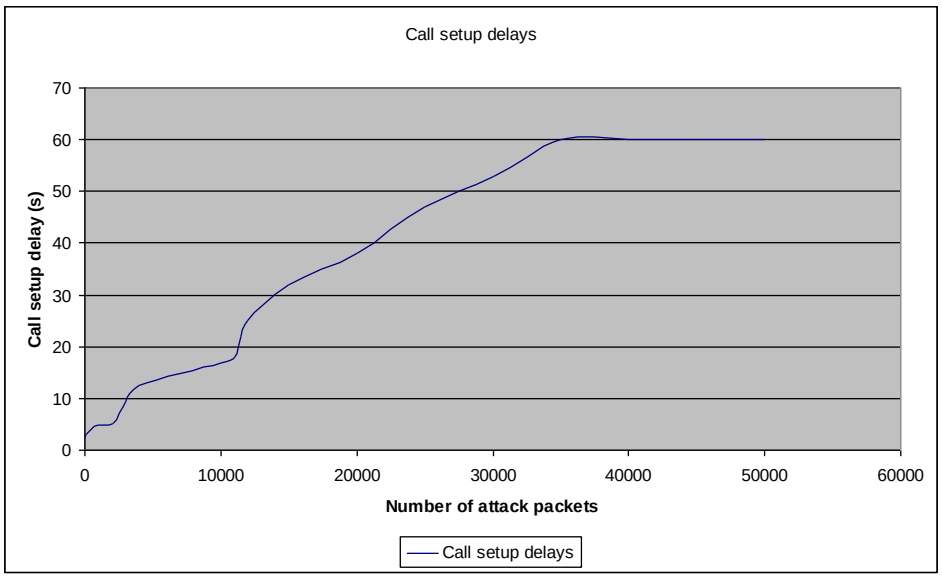


Figure 10: call setup delay during INVITE flood

From this experiment we can see that as the number of attack packets increases the call setup delay increases. When the amount of attack packets reaches a critical point, call setup will be timed out (the timeout configured for this testbed is 60 seconds).

This experiment demonstrates that a SIP-based VoIP system is vulnerable to SIP request flooding attacks.

3.2 Existing SIP flooding attack mitigations

3.2.1 Firewall

Implementation of a firewall is the most common security technique used to protect network components from external attacks.

Traditional firewalls use layer-3 filters to block unwanted traffic while some modern firewalls use application-layer gateways based on layer-7 filtering. Firewalls are generally designed for general purpose traffic filtering, and will often not detect application-specific attack traffic.

A series of tests were carried out to verify the effectiveness of firewall mitigation. The experiment testbed setup with a firewall is shown in figure 11. Five windows XP professional computer with 256MB ram are used to build this testbed, where three computers have X-lite 3.0 [2] installed are used as SIP users (two internal and one external), one computer is used as the attacker and one is used as the firewall and the packet analyser. Packet analysing tool we used is wireshark.

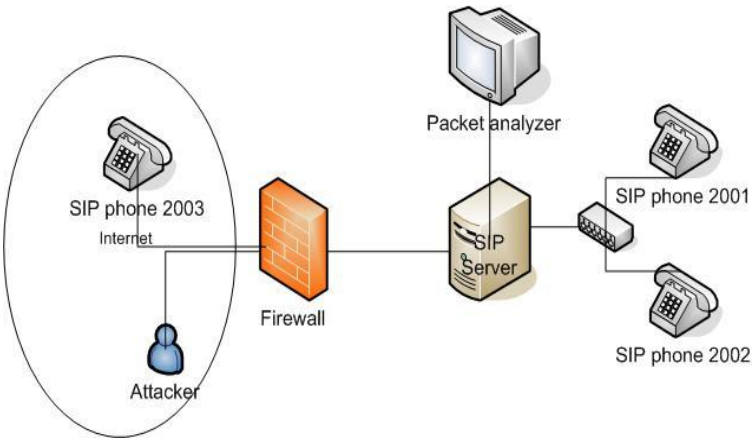


Figure 11: Firewalled Test Bed

3.2.1.1 Experiment set 1: WatchGuard Firewall

In the first set of experiments, a standard WatchGuard firebox 5 was used. WatchGuard firebox's external interface only accepts requests belonging to the same subnet. In our experiment, we assume that the attack knows the IP address range of the subnet. This is because it is not difficult to find out the address of the external

[2] X-Lite is a SIP soft phone developed by CounterPath Corp. http://www.counterpath.com/x-lite.html&active=4

interface of the firewall. For example, trace route can be used to find the address of that interface; additionally, if an attacker monitors the traffic of a legitimate user, it is not hard to guess the range of IP addresses used in this subnet.

There is an option on WatchGuard firebox to defeat DoS attacks, called “block spoofing attack”, which was supposed to be able to recognize packets with spoofed IP addresses and block them. In our experiment, we have enabled this function, and flooded the SIP proxy server with 60,000 INVITE requests. Figure 12 shows the attack command.

```
[root@testbed34]# ./iflood eth0 2002 opencloud.com 10.0.0.1
60000 ranged –S 2004
Enter starting IPv4 address: 10.0.0.2
Enter ending IPv4 address: 10.255.255.254
Sent: 24782
```

Figure 12: iFlood attack tool command

The call setup delay and the number of attack packets passing through the firewall were measured during the attack. Figure 13 shows the call setup delays during this attack, followed by a graph showing the number of attack packets passed through the firewall (Figure 14).

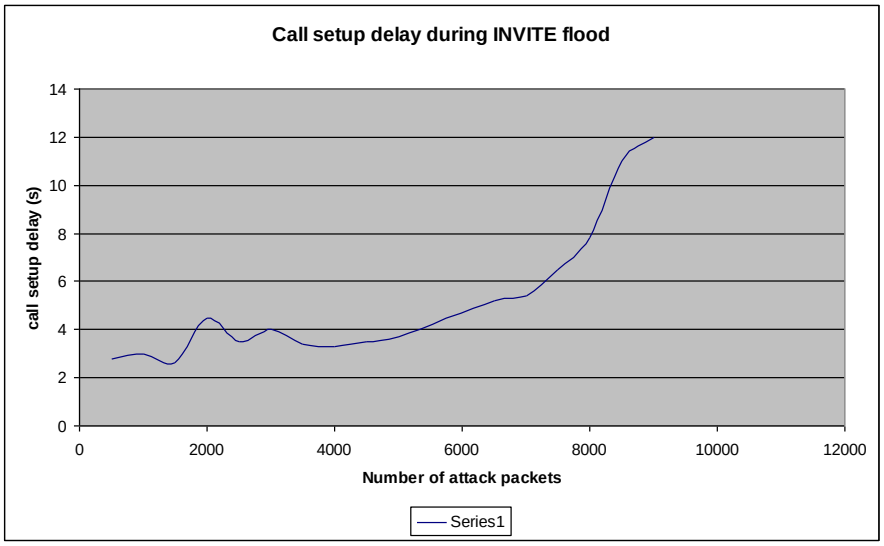


Figure 13: Call setup delay during INVITE flood

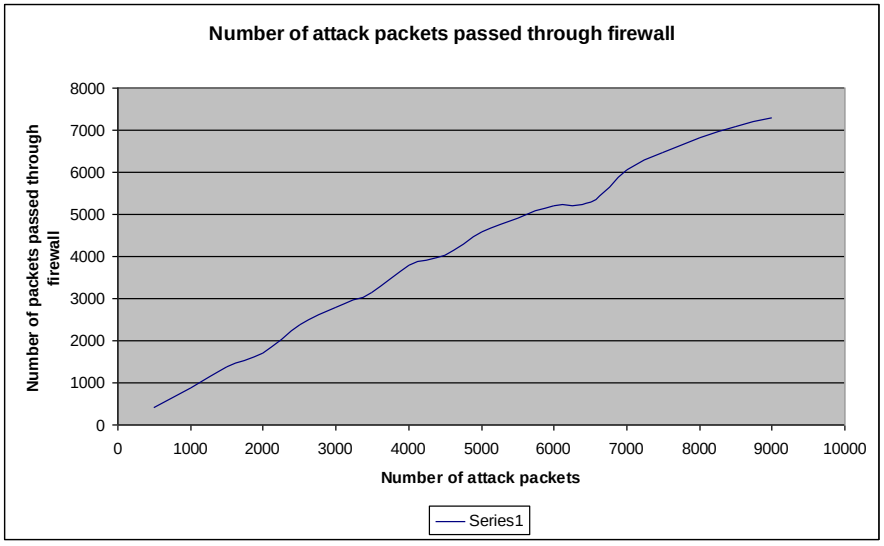


Figure 14: Number of attack packets passed through the firewall.

From this diagram, we can see that most of the spoofed INVITE flood can still pass through WatchGuard firebox even with the anti-spoof attack function enabled. Figure 13 shows that as the number of attack packets increases, the call setup delay increases. When the number of attack packets exceeds 8000, the VoIP service is almost unusable. The client starts to get “500-server internal error” responses.

When the number of attack packets exceeds 10,000, the percentage of "server internal error" responses was 83%.

This experiment shows that even with modern firewalls, SIP flood cannot be countered. In the next section, we use an intelligent SIP-capable firewall to mitigate SIP flood attacks.

3.2.1.2 Experiment set 2: AR450 Firewall

The second set of experiments was exactly the same as the first, but with the WatchGuard firewall replaced by the Allied Telesis SIP-aware AR450 firewalls.

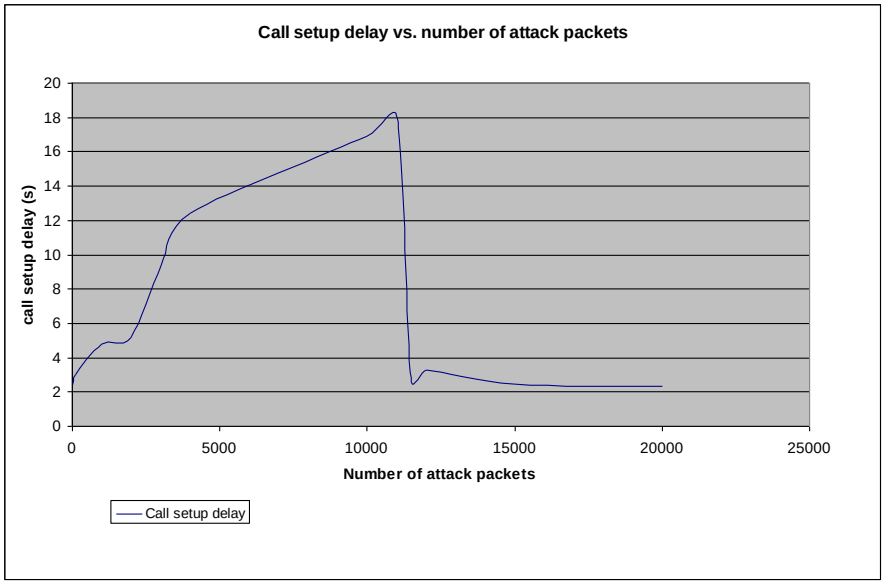


Figure 15: Call setup delay in SIP system when in flood burst mode

Figure 15 shows the client call setup delay with respect to the number of attack requests sent. The long delay in call setup should be partially caused by network link congestion.

Figure 16 shows the number of attack packets received with respect to the number of packets sent.

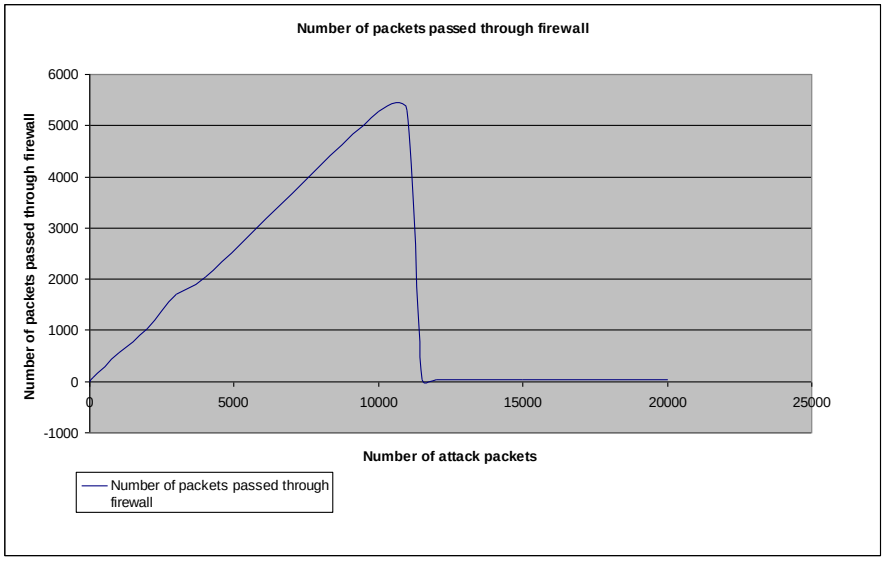


Figure 16: number of attack packets received on the SIP proxy server

As with the first set of tests, the call setup delay increases as the number of attack packets increases. This is because as more attack packets reach the SIP proxy server, less processing power is left for legitimate users, thus delay occurs.

However, in this experiment when the amount of attack traffic reaches a threshold, the firewall will detect the DDoS attack and block the flood traffic, and yet, still allow legitimate traffic to go through. After a series of test floods, the threshold value was found to be approximately 11,000 packets. However this value varies depending on the profile of previous attack traffic.

This DDoS attack traffic block behaviour is very similar to a router attack traffic traceback mechanism [65] [66]. In a router IP traceback mechanism, when a DDoS attack is detected, traceback is triggered. It would take a while for the router to determine the source of the attack.

3.2.1.3 Experiment set 3: improved iFlood

From the second experiment, we hypothesise that if the attack source generates short bursts of attack traffic, the firewall might not activate its defence mechanism. In order to test this hypothesis, an improved iFlood was developed. The improved iFlood adds an additional function which allows an attacker to optionally send the attack traffic in user sized chunks, with a specified delay between each chunk.

For the third experiment we used the AR450 firewall testbed and the improved iflood using a rate of 1000 packets per chunk, and an inter-chunk interval delay of one second.

Figure 17 shows the improved iFlood command.

```
[root@testbed34]# ./iflood eth0 2002 opencloud.com 10.0.0.1
60000 ranged  chunk –S 2004
Enter chunk size: 1000
Enter time delay: 1
Enter starting IPv4 address: 10.0.0.2
Enter ending IPv4 address: 10.255.255.254
Sent: 6000
```

Figure 17: usage of improved iFlood.

This specific chunk size and inter-chunk delay were arrived at based on intensive trial runs with varying sizes and delays. During these trials, we found that if the attack rate is too low, there would be little impact on the performance of the system. If the attack rate is too high, most of the attack traffic will be lost owing to the network congestion. With 1000 packets per chunk delay of one second, we are able to block all incoming calls, and with a packet loss rate. Figure 18 shows the call setup delay in this attack.

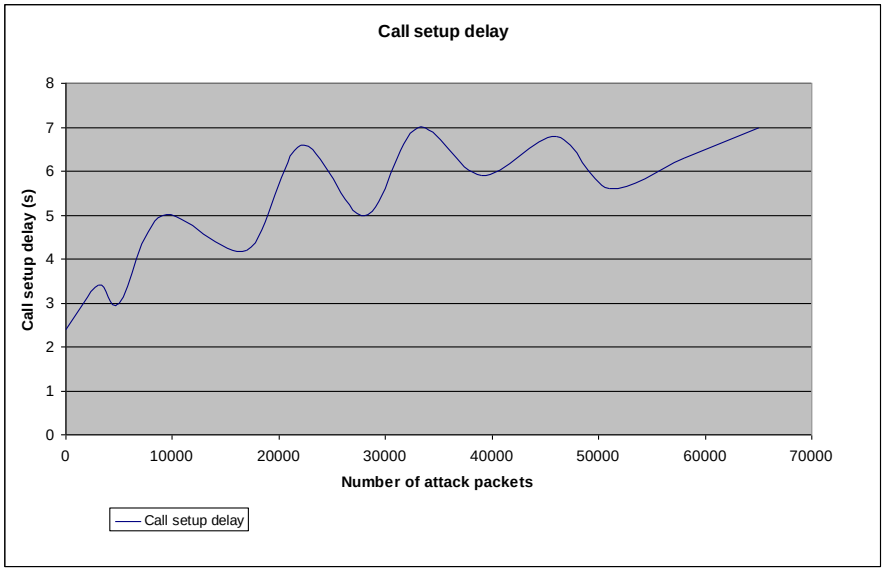


Figure 18: Call setup delay in chunk attack with a rate of 1000 packets per second

As the number of attack packets increases, the call setup delay increases. Monitoring showed that in this experiment 90.8% of the attack traffic passed through the firewall.

Figure 18 proves the hypothesis that by having short bursts of attack traffic, the attacker is able to penetrate the protection of the firewall with sufficient attack packets to cause a SIP denial of service. This implies that any firewall that implements similar security mechanisms can also be defeated by advanced SIP flood attacks.

3.2.2 Router-based flood mitigation

Being the intermediate nodes of the network, routers may be used to reduce the impact of flood-based DoS attacks. The router mitigation mechanisms can be categorized into three types: attack early detection, attack traffic filtering and attacker traceback.

3.2.2.1 Attack early detection

One type of router-based flood mitigation involves packet filtering, or bandwidth limitation on the intermediate routers [67] [57]. This mechanism usually requires specific agents to be installed on intermediate routers. Kashiwa, et al. [57] propose an Active Shaping mechanism to mitigate DDoS attacks which involves using extra monitoring and management components at the routers. At the root router near the protected network, a Probe Active Component (PAC) is used to monitor the traffic targeting at the protected network. If a DDoS attack is detected, the PAC sends messages to the Traffic-control Active Component (TAC) to shape the incoming traffic. TACs are implemented on all the routers.

Another example of traffic filtering is the Source Address Validity Enforcement protocol (SAVE) [68]. This protocol propagates source address information from the source location to the destination. SAVE runs on individual routers and builds *incoming tables* for them, allowing each router to verify whether each packet arrives at the expected interface. The router needs to save a large list containing the source address and destination prefix.

Router-based flood mitigation has the potential to stop attacks at an early stage and thus minimize the effect of the attacks. However, these approaches require ubiquitous adoption of the proposed standards and coordination among different routers and networks, making implementation difficult, and this approach impractical.

3.2.2.2 Attack traffic filtering

This operation requires a router to inspect the packets as they pass through. If a packet is not legitimate, the router should drop it. The two most common examples of this operation are ingress filtering

[54] and egress filtering [69]. Ingress and egress filters determine whether a packet is legitimate based on the source IP address of the packet. Ingress filters filter a certain range of IP addresses at the router's external interface. Egress filter only allows packets with IP addresses from its own subnet to be processed.

While attack traffic filtering can reduce the potential for flooding attack, it typically only eliminates attacks from certain network addresses, and spoofed flooding attack traffic is still able to pass through the routers. Again, to be effective, this requires that all routers adopt the protocol.

Peng et al. [70] propose a DDoS mitigation mechanism using history-based IP filtering for edge routers. In this approach, the edge router creates an IP address database which stores the source IP addresses of legitimate users, so when the system is subsequently under a DoS attack, the legitimate user traffic can be protected. The approach is as follows:

- distinguish legitimate traffic from attack traffic by using an IP address database;
- build an efficient lookup mechanism;
- apply filtering based on successful lookup.

The history-based approach can be useful to VoIP, as people tend to make phone calls to the same destination and this is more effective in VoIP than any other IP-based applications. Peng et al [70] argue that in a flash crowd event (a sharp and often overwhelming increase in the number of users), 82.9% of the IP addresses have appeared before. However, in a flood-based DoS event it has been reported that only 0.6-14% of the IP addresses have appeared before.

However, this approach does not address the fact that owing to the use of DHCP, the source IP addresses change over time, which

might cause an excessive amount of useless IP addresses being stored at the router, and slows the lookup process down.

3.2.2.3 Attacker traceback

Attacker traceback is an advanced security mechanism. There have been a number of proposals for traceback mechanisms to mitigate DoS attacks [71-74]. The simplest form of attacker traceback is IP traceback, which is concerned with detecting the source(s) of a DoS attack. However, since attackers often use spoofed IP addresses, it is impossible to use effective detection via a simple analysis of the IP header of the received packets. To avoid this problem, packet marking techniques can be employed [75]. The easiest form of marking is node append, where every router on the path crossed by a packet adds its IP address to the packet to facilitate the traceback process.

Attacker traceback is able to choke the attack traffic at the origin, so this approach is able to eliminate the impact of DoS attacks totally. The draw back of this approach is that it is difficult to implement and may introduce high overheads. Furthermore, this approach requires coordination among all intermediate routers along the network.

3.2.3 SIP intrusion detection

SIP intrusion detection has been studied by many researchers [76] [57] [77] [78] [79] [80] [81] [82]. This involves having a detection component to distinguish a SIP flooding traffic from normal SIP requests. The advantage of SIP intrusion detection mechanisms is that they do not typically require collaboration of a large number of hosts, which makes the implementation easier.

The most commonly used techniques in SIP intrusion detections are: a state machine-based detection engine, and a request header examine engine. An example of a request header examination is to use

hop-count information; the other technique is to use the attack traffic profile to identify the difference between attack traffic and normal traffic, thus limiting the attack traffic.

3.2.3.1 Use of a finite-state-machine to identify a SIP flood attack

H. Sengar et al. [80] proposed a VoIP intrusion detection mechanism through an interacting protocol state machine. In this approach, a finite state machine is used to record the status of the current SIP message transaction. An attack is detected if the SIP request received is not expected. Figure 19 shows the basic concept of this approach.

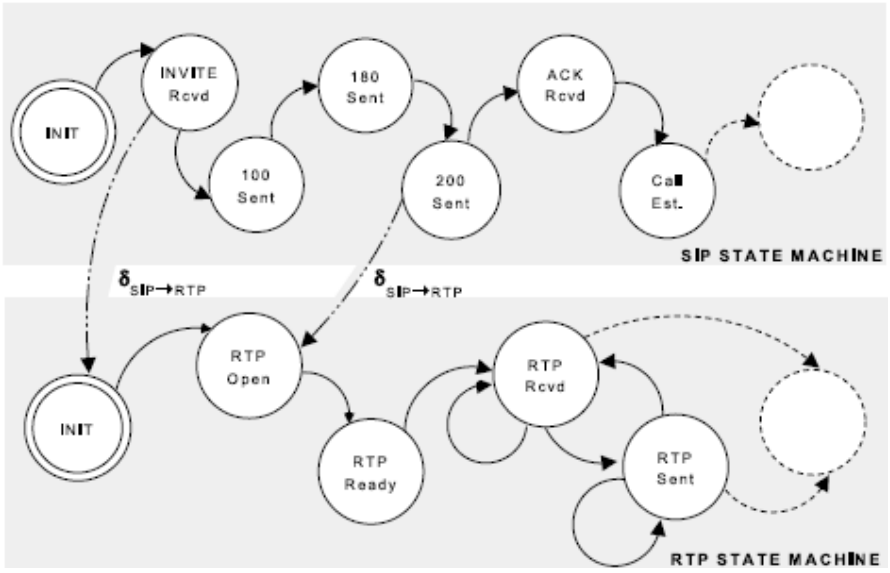


Figure 19: use of a finite state machine to identify SIP flood attacks

This approach is effective because SIP message flows have certain patterns. Any flow that does not follow the pattern is recognized as attack traffic. Every time a new session request is

received, the intrusion detection engine initiates a new flow pattern for it. This process is very computationally intensive. Chen [81] also proposed a similar but simpler approach to detect DoS attacks on the SIP system in which the intrusion detection engine checks each incoming SIP message and, if it has a new session ID, the engine will increment its error count. When the error count reaches a threshold, a flooding attack is assumed. When an attack is detected, the system can generate temporarily unavailable responses to incoming requests.

A finite state machine is complicated to implement and, since each state of the single session would be monitored and recorded, it consumes a significant amount of computational and memory resource which would make the system more vulnerable to flooding attacks. Furthermore, this approach can help to detect a SIP flooding attack, but it cannot reduce the effect of a SIP flooding attack.

3.2.3.2 Use of Hop-count information to identify illegal SIP requests

Hop-count information resides in the IP header, which is used to prevent endless circulation of IP packets. The time-to-live (TTL) field in the IP header specifies how many hops this packet is allowed to travel. Whenever the packet passes through a router, this value is decremented. When this value reaches 0, this packet will be dropped. Thus, the TTL fields directly indicate the distance of the source host. Haining Wang et al. [56] proposed a novel solution to mitigate spoof IP packets attacks based on the hop-count of incoming IP packets and their source IP address. In this approach, the router that is one hop away from the application server is in charge of checking the hop-count for all incoming requests. That router would firstly build a hop-count table, containing hop-count information on all possible destinations. For example, if a host from network 192.168.1.0/24 is 8

hops away from the server, Hc = 8. When an incoming request is received, the router first looks at its TTL and network address. If this network address has the same TTL as in the table, the request is processed, otherwise it is dropped. Figure 20 shows the hop count check algorithm.

```
For each packet:
  Extract the final TTL Tf and the source IP
address S;
  Compute the hop-count Hc = Ti –Tf;
  Index S so get the stored hop-count Hs;
  If (Hc not equal to Hs)
    The packet is spoofed;
  Else
```

Figure 20: Hop-count check algorithm.

This algorithm has been shown to be capable of discarding 90% of the spoofed IP packets [56].

You et al. [79] proposed a fast DDoS attack detection based on checking the TTL value in order to spot abnormal spikes on the incoming traffic. In this approach, all traffic that goes to an application server should have TTL with normal distribution. However, in the flooding attack scenario, attack traffic is likely to be generated by a single host and so the TTL is the same from all incoming packet. Thus, by monitoring the hop counts of incoming request, flooding attack would be detected. This approach is very simple, yet effective.

In order to generate a complete hop number table, thousands of addresses and hop counts have to be stored. When the number of entries increases, it becomes more difficult to find a particular network hop number. Additionally, hop count information is only accurate in

connection oriented connections for example, TCP. However, most of the VoIP traffic is carried by UDP, thus it is not very useful any more.

3.2.3.3 Use of a traffic profile to identify SIP flood traffic

Reynolds et al [78] proposed a multi-layered protection for IP telephony. This approach is based on the theory that a SIP INVITE request would finally trigger an OK response, thus in the long run, the total number of INVITEs received by the SIP proxy server, should be similar to the number of OK messages. In this approach, an application layer attack sensor is implemented to detect a SIP DDoS attack. The sensor is used to record the number of INVITE and OK messages from each URI. If the number difference between the pair is too large, DDoS attack is detected, and a "service temporarily unavailable" is generated to the host. This approach is too simple, and rather than preventing them can be easily used to cause spoofed DoS attacks on individual hosts.

Fowler et al. [77] propose a DDoS defending mechanism in an MPLS-based wireless network. In this approach, the pushback mechanism during congestion is used to identify a malicious host. However, this DDoS detection only works if the attackers use real IP addresses. If the DDoS packets use spoofed IP addresses, it is impossible to spot the attacker. An interesting aspect of this approach, however, is that multimedia traffic was given higher priority; this demonstrated that queuing is helpful to reduce the impact of a DDoS attack.

Overall, while a SIP intrusion detection mechanism is able to inform a system administrator when an attack has been detected, it will already have had an impact on the SIP system. Also, current detection mechanisms are either too complicated, requiring a lot of

computational resources (for example, a finite state machine), or too simple and hence ineffective.

3.2.4 SIP flood prevention

Instead of attempting to counter a DoS attack after its detection, a better approach is to prevent the occurrence of SIP flood attacks in the first place. Attack prevention is said to be one of the most effective defence approaches for DoS attacks that use spoofed traffic [83]. In this section, we will discuss two effective SIP flood prevention mechanisms: the predictive-nonce approach [84] and layer-3 queuing [85].

3.2.4.1 Predictive-nonce for mitigating SIP flood

As mentioned earlier, using SIP digest authentication can make a SIP proxy more vulnerable to SIP flooding attacks as a result of the need to use RAM to store the generated nonce. Rosenberg et al. [84] propose a predictive nonce (p-nonce) solution to overcome this weakness. This approach proposes that the proxy server should generate a nonce based on the SIP header fields that do not change within the same session. The nonce is generated through a cryptographic secret function over the session's unique field. Figure 21 illustrates the SIP predictive nonce challenge process.

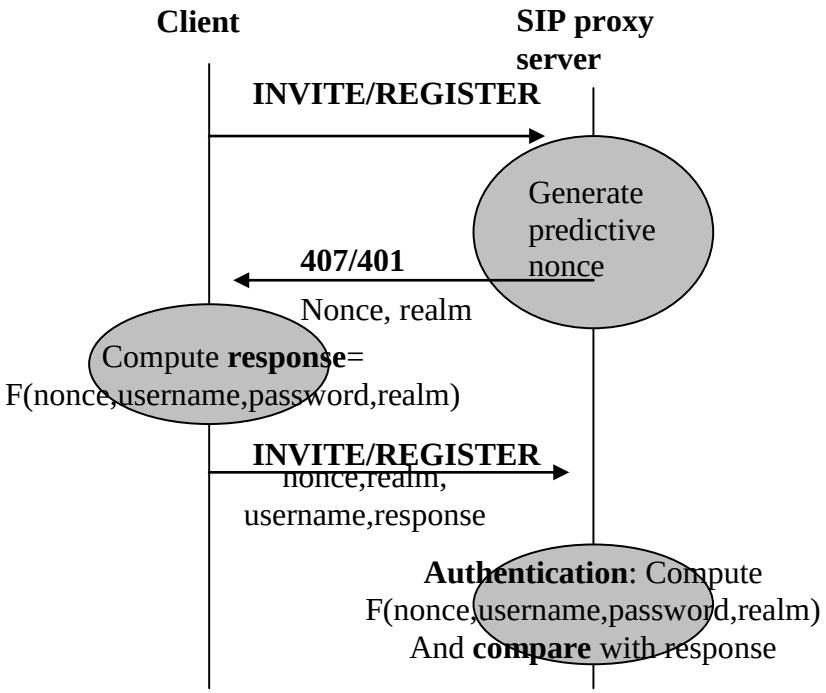


Figure 21: Process of SIP predictive nonce checking

When the request with the authorization header arrives, the server recomputes the nonce using the same set of headers in the same way. If the headers have not changed, the resulting nonce will be identical to the one issued in the challenge, and the digest response will be valid. If any of the header fields have been changed by an attacker, the nonce that is computed will be different, the server will detect this condition, and the request will be rejected. This approach can be used to eliminate spoofed SIP flooding traffic, as the attacker using spoofed source IP addresses will not be able to receive the nonce.

The advantage of this approach is that it is able to prevent SIP flooding attacks. This approach enforces the use of the three-way handshake, which means spoofed SIP flooding attacks will not succeed. Furthermore, it does not occupy the scarce RAM resource on the SIP

proxy server to store the generated nonce value. Thus it should provide a better performance than the traditional SIP authentication process.
The drawback to this mechanism is that the approach requires authentication for each request, and the computation process is very intensive, as each verification process requires duo-computation (one to calculate nonce, one to calculate the response). In order to achieve the level of throughput on a traditional authentication-enabled SIP system, the SIP proxy server has to have higher processing power.

3.2.4.2 Queuing mechanism to prevent flooding attacks

Various researchers [77] [85] [86] have shown that the effect of a flood-based DoS attack can be reduced if the system has a good queuing mechanism.

Ohta [85] studied the performance of a SIP signalling network in an overload condition and proposed improving the performance of the network by implementing a better queuing mechanism. In this approach two FIFO queues are implemented, one to handle SIP INVITE requests and the other is used to handle all other messages. The SIP INVITE queue was given a lower priority. The performance of a single FIFO queue and a two class priority queue are compared using Network Simulator 2. The research demonstrated the performance of the network under the two scenarios, and verified with a two class priority queue, that the performance of the system was improved.

The most significant advantage of layer-3 queues is their high speed because there is no application processing involved. Additionally, they can be implemented on the firewall relatively easily thus offloading work from the SIP server. However, this approach is too simple as it just distinguishes and assigns lower priority to SIP

INVITE requests. This makes the system more vulnerable to other flooding attacks. For example, if an attacker floods the server with ACK packets, then the call setup time for the legitimate user will be increased.

3.2.4.3 Two layer DoS prevention on the SIP VoIP infrastructure

Ehlert et al [87] proposed a SIP DoS solution, which consists of two main components: an IDS enabled firewall; and an enhanced SIP proxy server. This system is designed to defeat SIP flooding, malformed message attack as well as DNS DoS attack. The IDS monitors incoming traffic and triggers an alert if the incoming traffic reaches a threshold (e.g. 100 INVITEs per minute). In this case the cache only answers requests from its stored content, and returns an unresolvable message for any request that cannot be answered directly from the cache. The SIP proxy server has a module built-in to examine the content of a SIP message header to spot malformed SIP requests.

This system is ineffective if the flood packets are well-formed and use randomly spoofed source IP addresses.

3.2.5 SIP flood mitigation summary

The existing SIP flood mitigation provides some partial solution for SIP DoS attacks. However, they all have their own limitations and can hardly be used as a final solution for SIP DoS attacks. Table 3 summarises the advantages and disadvantages of all these solutions.

Table 3: Existing SIP flood mitigation techniques

Mitigation	Advantages	Disadvantages

Technique		
Firewall	• Relatively easy to implement; • Firewall avoids the necessity of having to rely on user cooperation and responsibility.	• Cannot protect against attacks from internal network; • Our experiments shows the security on firewall can be defeated;
Router-based approach	• Has the potential to stop attacks at an early stage, thus minimizing the effect of them; • Some techniques help to eliminate the impact of DoS attacks totally (attacker traceback);	• Requires ubiquitous adoption and coordination among different routers, which makes the implementation difficult; • Some approaches requires coordination among all intermediate routers along the network, which makes them impractical (attacker traceback); • Some approaches would cause an excessive amount of useless information to be stored at the router and slow the lookup process down;
Intrusion Detection	• Helps to spot attack traffic at real-time, so further anti-flood mechanisms can be triggered to stop the attacks;	• It does not provide mechanisms to stop the attack traffic; • Normally when the attack is detected, it is too late already; • This is only a partial solution to SIP flood attacks; • Some intrusion detection mechanisms can be very complicated and require a large amount of computation power (finite state machine approach) • Some require the router to store a large amount of information, which can slow down the matching process (hop-count approach);
Intrusion Prevention	• This is considered to be the most effective SIP flood mitigation approach; • Some approaches can	• Some of the approaches are very processing intensive and are relatively

	eliminate the impact of SIP flood attacks (predictive nonce approach); • Some approaches can reduce the impact of SIP flood attacks (layer-3 queuing approach);	slow (predictive nonce approach); • Some approaches just provide partial solutions; • None of the existing approaches can help to eliminate the impact of SIP flood attack, while maintaining a good QoS for legitimate users;

Chapter 4: Security-enhanced SIP system (SESS)

In this chapter we propose a security-enhanced SIP system (SESS) consisting of a security enhanced SIP proxy server and an enhanced application layer stateless firewall [64]. The basic concept of SESS is to maintain in both the firewall and the SIP server the addresses of known (legitimate) users in order to give them priority handling. The enhanced SIP proxy server updates the firewall with the IP addresses of legitimate users and alerts the firewall when a legitimate user IP address expires and should be removed from the list. An enhanced firewall adjusts its rules according to the information fed by the enhanced SIP proxy server, and enforces advanced predictive nonce checking on unknown users. Additionally, a new protocol called Known Address Synchronization Protocol (KASP) is proposed to manage the update of legitimate user information between the security enhanced SIP proxy server and the reactive firewall.

4.1 Related work

Ohta [85] has studied the performance of a SIP signalling network in an overload condition and proposes a mechanism to improve the performance of VoIP by giving the INVITE message lower priority. The performance of a FIFO queue and a two class priority queue are compared. However, this queuing mechanism is simple, and it does not protect the system from SIP flooding attacks.

Studies have shown that to a web service the difference between a very busy day and a DDoS attack is that in a very busy day, 82.9% of the IP addresses have appeared at the web site before. However, in a DDoS attack event only 0.6-14% of the IP addresses have appeared before [88]. We can assume this statistic would be more extreme on a

VoIP server, because people tend to call the same destination over and over again. Researchers [89] [70] have proposed history-based IP filtering to mitigate flooding attacks. Peng [85] considers an IP address to be legitimate if it can satisfy two rules: it appeared for '*d*' days, and there are '*n*' packets transmitted using this IP address. When the edge router is overloaded, it will only admit "legitimate" packets through. The advantages of this approach are that the scheme is robust, does not need the cooperation of the whole Internet community, is applicable to a wide variety of traffic types and requires little configuration. Furthermore, it uses a list to store legitimate user addresses, so when the system is under a DDoS attack, the service performance of legitimate users can be guaranteed. However, when the system is overloaded, packets whose IP addresses that are not on the legitimate user list will be dropped, resulting in the risk that a legitimate user could be refused service.

D'Souza et al. [90] propose a method to mitigate spoofed packet attacks which also takes advantage of an IP history. In their approach, a packet classification engine is used to match the source IP addresses of the incoming packets with a list of "known" hosts. The "known" hosts are then queued to a higher priority queue. After an "unknown" host is authenticated, it will be put in the trust list, and its packets will be promoted into the higher priority queue. D'Souza does not explain how "known" traffic is determined.

SESS extends and integrates the solutions proposed by Peng and D'Souza, resulting in a system which consists of three components: a security-enhanced SIP server, a security-enhanced firewall, and a new protocol called "Known Address Synchronisation Protocol" (KASP) which is used to carry the legitimate user notifications.

4.2 Overview of the proposed solution

In a SIP-based VoIP system, a "known host" based priority queuing mechanism can be very helpful in defending against DDoS attacks because:

- People tend to call the same destination all the time, which means the SIP proxy server is likely in any reasonable time period to receive requests from the same clients. By recording the legitimate source IP over a long period, the proxy server would have an almost complete list of legitimate users that would place a phone call.
- In SIP, it is reasonably easy to determine a valid user as the SIP call setup is a handshake process and it cannot be completed with spoofed IP. Thus, we can assume all IP addresses that have completed a handshake are legitimate, which includes all users that have successfully registered or made a phone call.

While the concept of a known user list will work for SIP, it would be reasonable to assume that application of the service would benefit from having more frequent users given higher priority for SIP service, especially when the server is under heavy DDoS attack. Consequently, we propose building on Ohta's work by implementing a dual-stage priority list.

The IP addresses of a SIP client host will normally be assigned by DHCP[91] and so may change. Consequently, we propose that the "known host" list should have an expiry time in order to remove potentially obsolete addresses and to keep the known host list at a manageable size. Further, the application of a protection mechanism using the IP list requires that the list be synchronised at both the firewall and the SIP server, and so we propose a new Known Address Synchronisation Protocol (KASP).

Figure 22 illustrates the overall operation of the security enhanced SIP system (SESS).

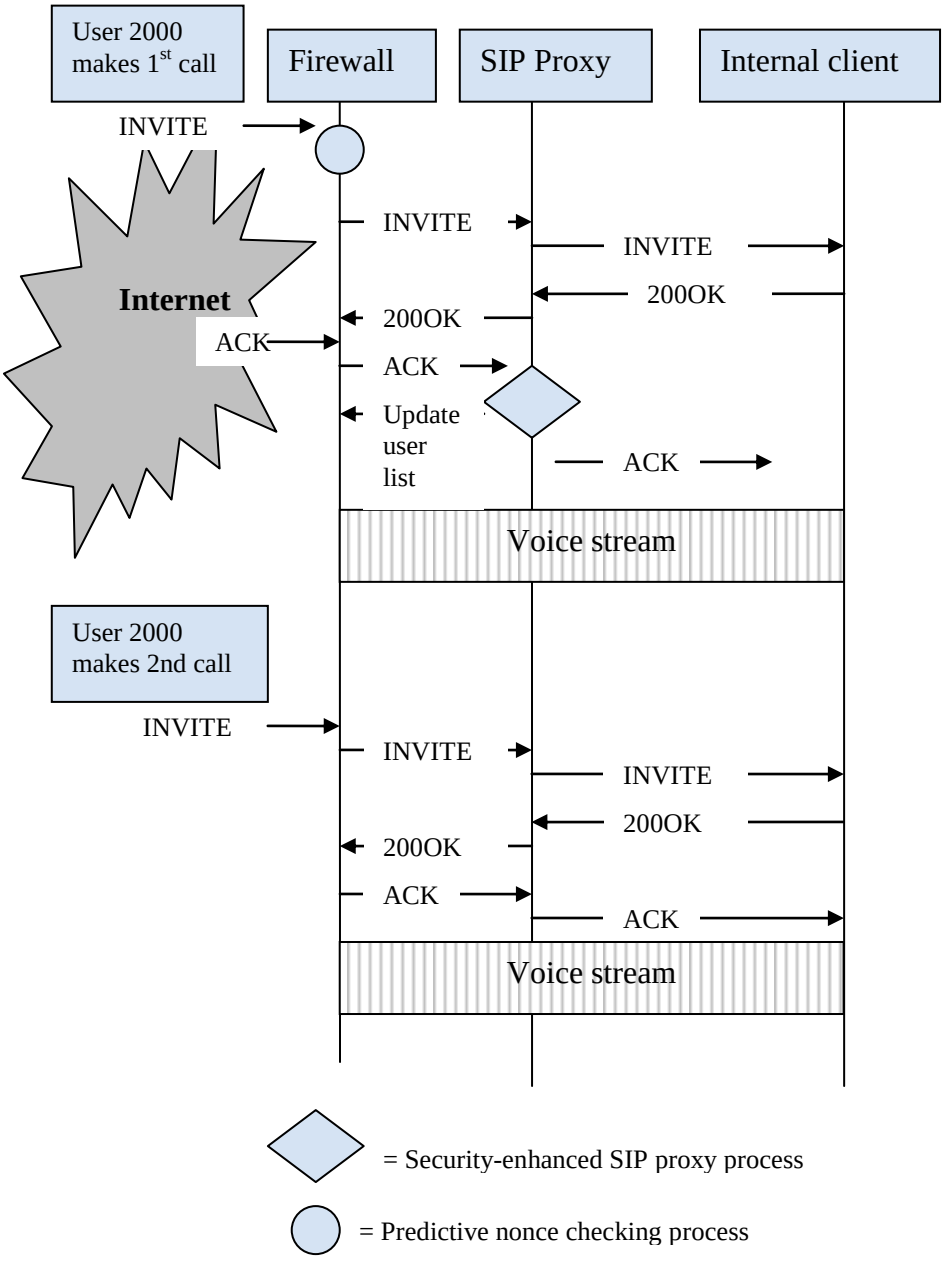


Figure 22: Overall process of Security-enhanced SIP proxy server

When a SIP request comes in at the firewall's SIP port (5060), it will check whether it is an INVITE or REGISTER. If it is, the firewall

will authenticate the source by using improved predictive nonce checking (this will be explained in section 4.4.1). If not, it will just forward the request to the SIP proxy server. After the sender is authenticated, the re-INVITE and re-REGISTER will be passed to the SIP proxy server.

The SIP proxy server then would proxy the request to the callee. When the callee picks up the phone, a 200OK response is sent back to the caller via the proxy server. As soon as the caller receives the 200OK response, an ACK request is generated to inform the SIP proxy server and the callee about the success of the session setup.

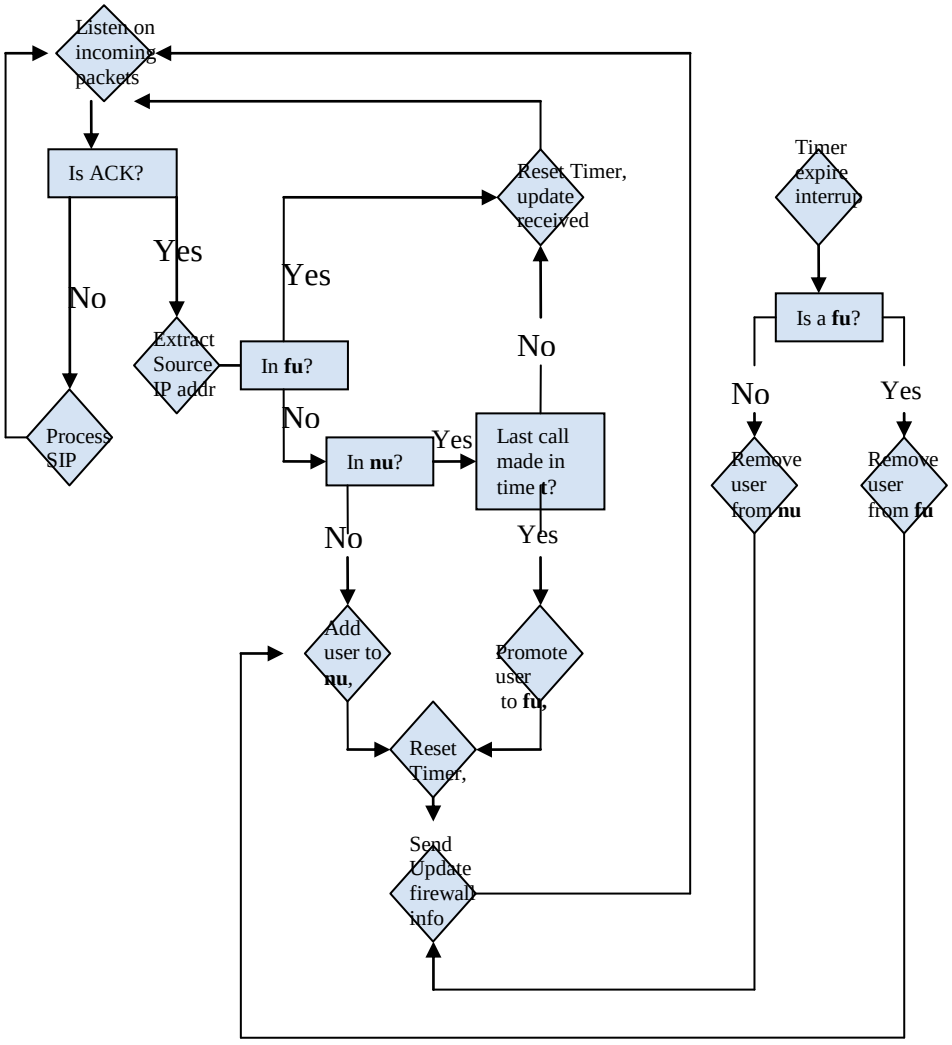
When the SIP proxy server receives the ACK request, it knows the call setup three-way handshake is finished. At this point it has been established that the caller is a legitimate user. Thus, the SIP proxy server will extract the source IP address and record it in the legitimate user list. Following this, the SIP proxy server will update the firewall with the changes on the legitimate user list.

After the firewall receives the updates on the legitimate user list, it will convert those updates to iptables rules sets and issue them to the kernel iptables module.

4.3 Security-enhanced SIP proxy server

The security-enhanced SIP proxy server is used to record the source IP addresses of legitimate users, send these addresses to the firewall, remove the users from legitimate user lists when the entry expires, and update the firewall with the expiry alert. The reason we have chosen to have the SIP proxy server record the IP addresses rather than the firewall is because the SIP proxy server would process SIP requests anyway and so recording the source IP field of a SIP message in an internal list would barely use any extra computational

power, so the performance of the proxy would not be significantly affected. However if extra application-layer processes are added to a firewall, the performance of the firewall would be much more significantly affected. Figure 23 illustrates the security operation on the security-enhanced SIP proxy server.



nu = userlist
fu = frequent userlist

Figure 23: security operation on the security-enhanced SIP proxy server.

As an extension to Souza[90], the proposed project uses a two-stage list instead of a single list. This allows legitimate users to be differentiated into two groups – occasional legitimate users, and frequent legitimate users who make more frequent phone calls. We suggest that frequent users should be assigned a higher priority.

Legitimate users are defined as users who have successfully placed a call. It is important to know that a successful registration does not guarantee the user is an authorized user, because registration is not a three-way handshake process.

As mentioned earlier, a call setup process is a complete three-way handshake and detection of an ACK request signifies the success of the process. Thus, when an ACK message is received, the proxy server can record the source IP address as a legitimate user. Legitimate user IP addresses are recorded into one of two lists (*userlist* and a *frequent userlist*) depending on how frequently the user uses the system.

When an ACK request is received by the proxy server, the proxy server will extract the source IP address of the request, and search through the *frequent userlist* and *userlist*, to see if this user is already on one of them. If this is a new user, the address will be added to the normal *userlist* and a timer will be set to remove the user on expiry after time ***t1***. A KASP message will then be sent to the firewall to notify the addition of a normal user.

If the user is already on the frequent userlist, the firewall will just reset the last-seen timer for that user.

If the user is on the *userlist*, the firewall will check the time difference between this and the previous call. If it is less than the frequent user expiry time ***t2*** the user will be promoted from the normal

list to the frequent userlist, and the firewall will be notified of the change.

If the user is on the userlist, while the time difference between this and the last call is greater than ***t2***, the proxy server will just update the timer.

When a user on the *frequent userlist* expires, it will be removed from the *frequent userlist*, and be added to the *userlist*. A KASP message will be sent to the firewall to update the changes.

When a user is expired from the *userlist*, it will be removed from the *userlist.* The next time this user makes a phone call via the SIP server, it will have to go through the predictive nonce checking process again.

4.4 Known address synchronization protocol (KASP)

KASP is used to transmit the updates of legitimate user lists from the SIP proxy server to the firewall using UDP. The reason to use UDP over TCP is that TCP is a connection oriented protocol which would need a three-way-handshake to establish a session before the data is transmitted. While TCP has better security than UDP, since the information update happens between a SIP proxy server and a firewall which is normally one hop away from the server, and given that the communication link is secured by the firewall, there is no real benefit to be gained from the security advantage of TCP. Additionally, as the update happens relatively frequently and the payload is small, it is very inefficient to go through the handshake process on each update.

Figure 24 shows the structure of a sample KASP message.

IP Header	UDP Header	KASP:+fu10.0.0.34

Figure 24: Sample KASP message

The first four characters in a KASP packet indicate this is a KASP message. The plus and minus indicate whether to add or remove a user, followed by two characters to indicate to which list this packet refers, and the rest of the message contains the IP address that is to be added/removed from the list.

4.5 Security-enhanced firewall

The security-enhanced firewall categorizes packets into three categories: frequent users, normal users and unknown users. Correspondingly, there are three queues on the firewall, namely: *high-priority queue*, *normal queue* and *suspicious queue*. Packets from hosts that are on the "*frequent user list*" will be put in the "*high-priority*" queue, if the packets are from a "*user list*", they will be put into the "*normal*" queue. And if the source IP of the packets is not on the lists, they will be put into the "*suspicious*" queue. Packets in the high priority queue will have the highest priority, followed by the normal queue, and they will be directly forwarded through the firewall to the server. Packets that are on the suspicious queue will be passed to the firewall's upper layer to be authenticated using the improved predictive nonce checking mechanism. In the following section, the improved predictive nonce checking mechanism will be discussed in detail.

In the initial approach, this project proposed an application-layer stateless firewall [64]. As an extension to the predictive nonce checking mechanism [84], we propose as an initial approach that the predictive nonce checking should be located on an application layer stateless firewall and the procedure should be simpler than that proposed in [84].

It is useful to put predictive nonce checking on the firewall because at a service provider level we can assume that the bandwidth at the firewall is normally much greater than that at the server, thus it is better to stop the flooding packets there before they reach the server. Furthermore, a proxy server is normally responsible for many tasks, e.g: callee address resolution, registration, SIP message routing and thus is generally subject to higher per-transaction processing loads than the firewall.

We propose an improved predictive nonce checking mechanism, which simplifies the predictive nonce checking process to avoid the firewall having to retain a copy of legitimate usernames and passwords.

4.5.1 Improved predictive nonce checking and the application-layer stateless firewall

This section details the improved predictive nonce checking mechanisms and the initially proposed application-layer stateless firewall.

There is a unique field (nonce) in the SIP 401 (Unauthorized) and 407 (Authentication Required) messages to avoid replay attacks. A nonce is a server-specified data string which should be uniquely generated each time a 401 or 407 response is made. The nonce is generated as a result of cryptographic function over some session-unique header fields plus other fields and a secret that is only known by the firewall, for example: callID, source IP address, and a secret. By doing this, the firewall does not have to record the nonces with corresponding callIDs. Next time, when the re-request message arrives, the firewall can recalculate the nonce on the fly based on the received SIP header fields. By comparing the recalculated nonce with the received nonce, if they match, the user is a legitimate user.

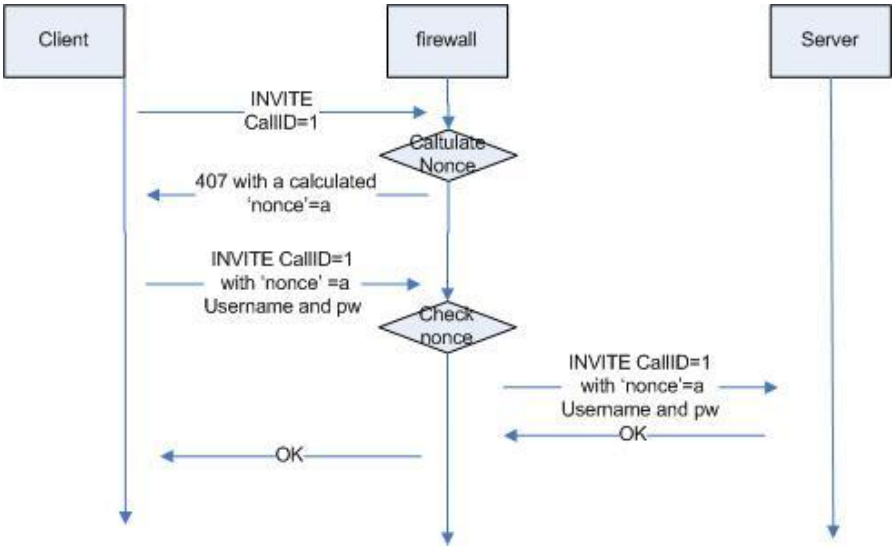


Figure 25: Call setup process with application-layer stateless firewall.

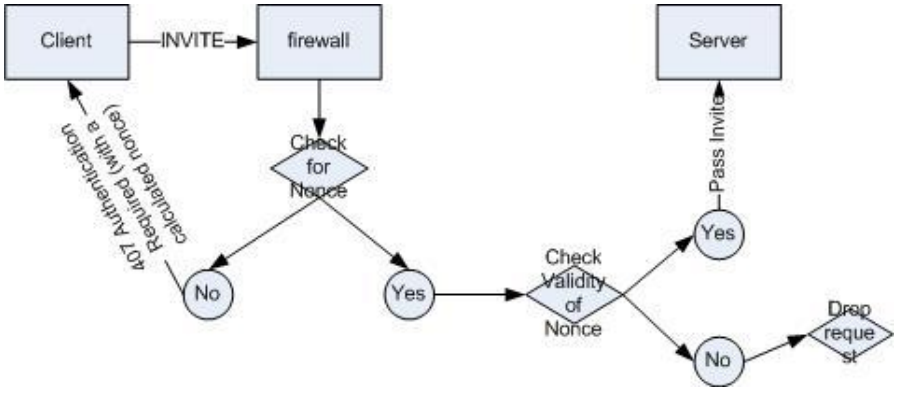


Figure 26: How an INVITE request is handled

Figure 25 shows the call setup process with firewall authentication using predictive nonce checking, and figure 26 shows the flow diagram of the handling of an INVITE message.

- When a message arrives at port 5060, the firewall checks if it is an INVITE or REGISTER message. If not, the message is allowed to pass through.

- If the message is INVITE/REGISTER, the firewall will check the SIP header, looking for a “nonce” value.
- If the incoming SIP request does not have one, the firewall generates a nonce value. This is the result of a cryptographic secret function computed over the CallID and source IP address which ensures that the nonce is unique for each session, as a new CallID is generated when a session is initiated. Figure 27 shows how the nonce is calculated.

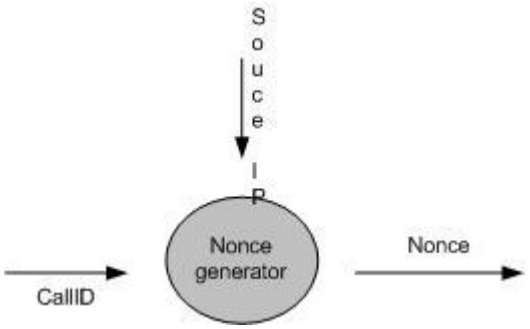


Figure 27: The generation of the nonce

- The firewall will then send back a 407/401 (Authentication Required/Unauthorized) message to the client, with the calculated nonce value. Then the firewall will drop the session.
- After the client receives a 407/401 message, it resends an INVITE message with the same CallID, server-specified nonce, username and password.
- When the firewall receives an INVITE with a nonce value, it will recalculate a nonce value based on CallID and source IP address and compare it with the received nonce.
- If the nonce matches, the request is allowed to pass through to the server.

- For better security, the cryptographic secret should be changed after a small period of time, e.g. 30 seconds. (an overlap for 2 successive secrets should be allowed)

It is worth noting that this firewall should pass all other SIP messages through in order to complete the session setup or teardown process.

4.5.1.1 Advantages

The advantages of the improved predictive nonce checking are as follows:

- It can protect the server from Spoofed SIP INVITE and REGISTER flooding attacks.
- Stateless authentication --The firewall does not need to store multiple CallID and nonce entries in a database. This protects the firewall from RAM exhaustion during a flooding attack.

4.5.1.2 Drawbacks

The drawbacks of the improved predictive nonce checking are as follows:

- There is a lack of kernel support as the predictive nonce checking is not native to the iptables firewall leading to slower than desired processing.
- The IPQueue module is used to pass the packets from the network interface to the application process, where the IPQueue is a single FIFO queue and the next packet is not passed until the first packet is processed. Consequently, in the event of a flooding attack, the call setup delay increases significantly.
- This approach is not able to block other SIP message flood traffic other than INVITE and REGISTER floods. However, since other SIP messages would not require much processing from the SIP proxy server, this is of relatively low concern.

4.5.1.3 Tests and results

In order to test how well this stateless firewall combats the SIP flooding traffic, and to verify our assumption above, a series of tests were carried out using the same testbed which was used for the previous firewalled experiments.

The firewall [64] used in this experiment is the stateless predictive nonce checking firewall. It is implemented in conjunction with my colleague Isaac Lee. Figure 28 shows the structure of this firewall.

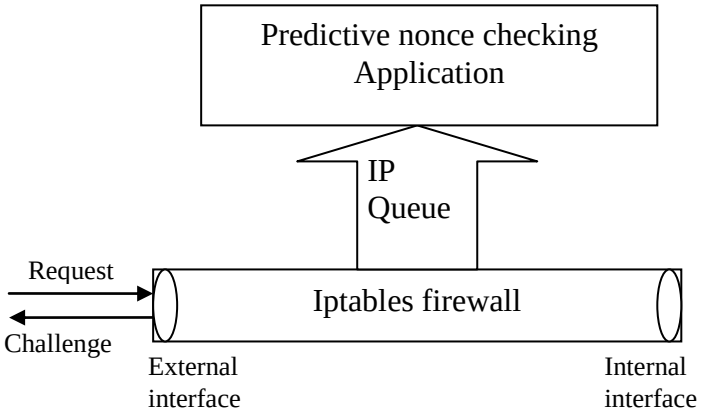


Figure 28: Predictive nonce checking firewall structure

It is based on an Iptables firewall, and uses IPQUEUE to pass each request from the network interface to the application-layer. It is important to note that IPQUEUE contains a single FIFO queue to pass all relevant packets to the application layer. Thus, when this system is under flooding attack, it might slow down the call setup process for legitimate users as well. After the packets have arrived at the application layer, the application layer application will process the request as specified in figure 24.

For the purpose of determining the optimum rate of flooding traffic, the maximum speed of the test bed network used in this experiment needs to be calculated. Initially, the call setup delay is monitored when the system is operating with the firewall but without nonce checking.

Then using the predictive nonce-checking firewall, the total number of received attack packets are recorded to calculate the percentage of the attack traffic that managed to navigate through the firewall and the client call setup delay was measured to verify the system performance impact of this firewall.

By comparing these two sets of experimental results, we can establish the efficiency of the application-layer stateless firewall.

We flooded the SIP proxy server, with no security, and verified with burst flood mode, that the average packets received by the server were 3229 packets per second. Thus, using 1000 packets per chunk with one second delays between chunks as our attack rate would be a suitable choice, as it provides a minimum packet loss rate.

Figure 29 illustrates the CPU usage diagram on the SIP proxy server, when using the application-layer stateless firewall.

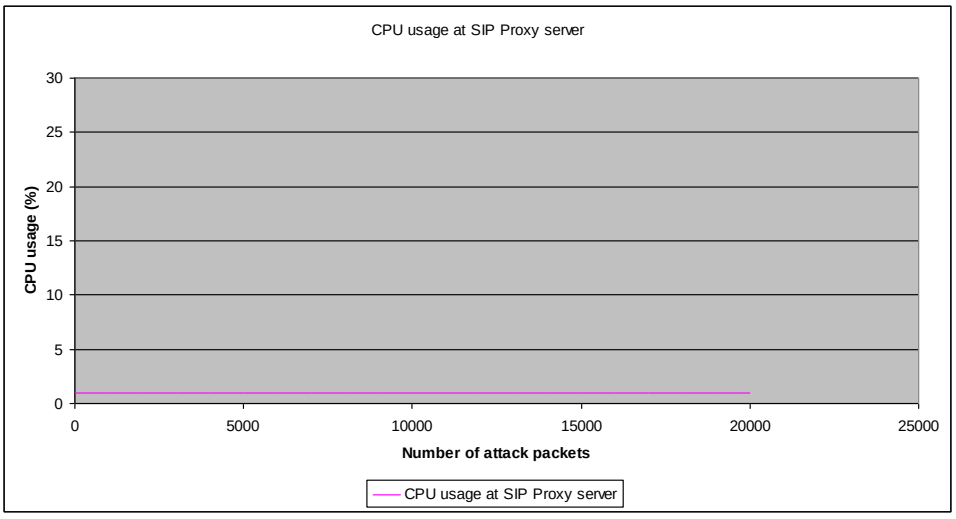


Figure 29: The CPU usage of the SIP proxy server during an INVITE flood attack

Figure 29 shows that during an INVITE flood attack the SIP proxy server is not affected. This verifies that the application-layer stateless firewall is able to block all spoofed INVITE and REGISTER packets. Figure 30 compares the call setup delays when the system is operating with the stateless firewall but without predictive nonce checking.

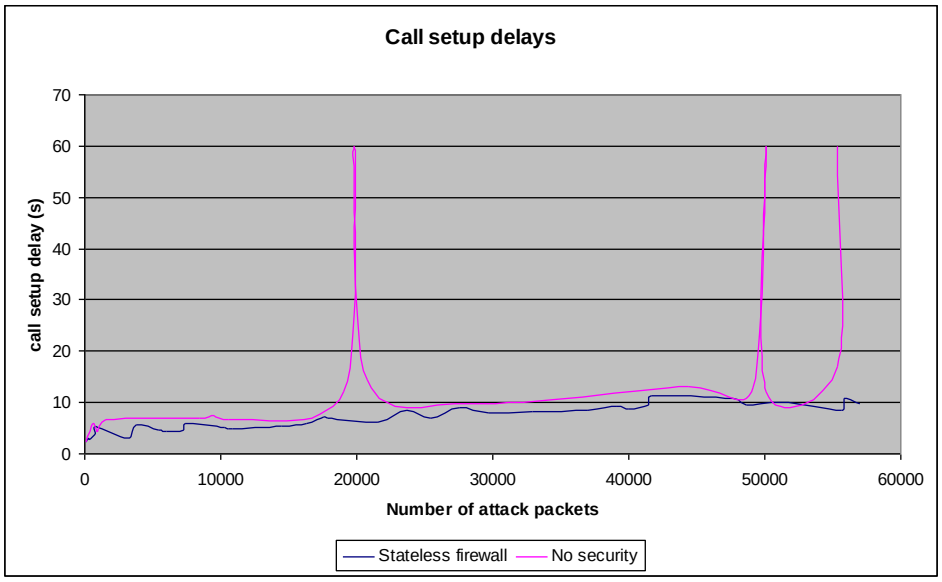


Figure 30: Comparison of call setup delays for stateless firewall and "no security" when the system is under INVITE flood at the rate of 1000 packets per chunk, one second delay.

The pink spikes in figure 30 represent call setup timeout. We can see the application-layer stateless firewall helps to eliminates call setup timeouts. This is because call setup timeouts are caused by SIP proxy processing power exhaustion. As there is no impact on the server, no call setup delays occur.

4.5.1.4 Analysis and Conclusion

From figure 30 we can see that the average call setup delay under an INVITE flood in an unsecured system is 9.39 seconds (call setup timeouts are not included in the calculation). This delay is caused by SIP proxy process delay. When using the application-layer stateless firewall system the average call setup delay is decreased by 25% (7.08 seconds), and this is mainly caused by the queuing and authentication process of the firewall.

While the use of application-layer stateless firewall can eliminate call setup timeouts and reduce the call setup delay by a quarter of that under no security, there is still a significant call setup delay. This confirms that the predictive nonce checking causes significant delays in the call setup process under flooding attack. This is because predictive nonce checking is not native to a system, so there is no kernel support, and a single FIFO queue is used to pass all SIP INVITE and REGISTER packets from the network interface to the application-layer process which makes the setup delay increase as the number of attack packets increases.

In order to achieve good service performance for a legitimate user when under a flooding attack, the proposed improved predictive nonce checking mechanism has to be used in conjunction with SESS.

4.6 Advantages and Drawbacks of SESS

The advantages of SESS are:

- **Good attack block success rate**: In theory, it is able to block all SIP INVITE and REGISTER floods.

- **High QoS for legitimate users**: Even when the system is under severe INVITE or REGISTER flooding attack, the QoS of legitimate users is still maintained at a good level.

The disadvantage of SESS is that it can only block INVITE and REGISTER flooding attacks. It does not have a way to block other SIP request flooding.

Chapter 5 of this document proposes an improved SESS (ISESS) which is able to block the majority types of spoofed SIP flooding requests, for example INVITE, REGISTER, ACK, and CANCEL floods etc.

4.6 Improved security-enhanced SIP system (ISESS)

As mentioned in last section, even though SESS is able to defeat SIP INVITE and REGISTER flooding attacks to provide a good QoS for legitimate users it was not able to block some other types of SIP request flood attacks. Consequently, an improved security-enhanced SIP system (ISESS) was designed which adds protection for all types of SIP requests flooding.

4.6.1 Overview of the improved security-enhanced SIP system

Figure 31 illustrates the process of this system, followed by a detailed explanation.

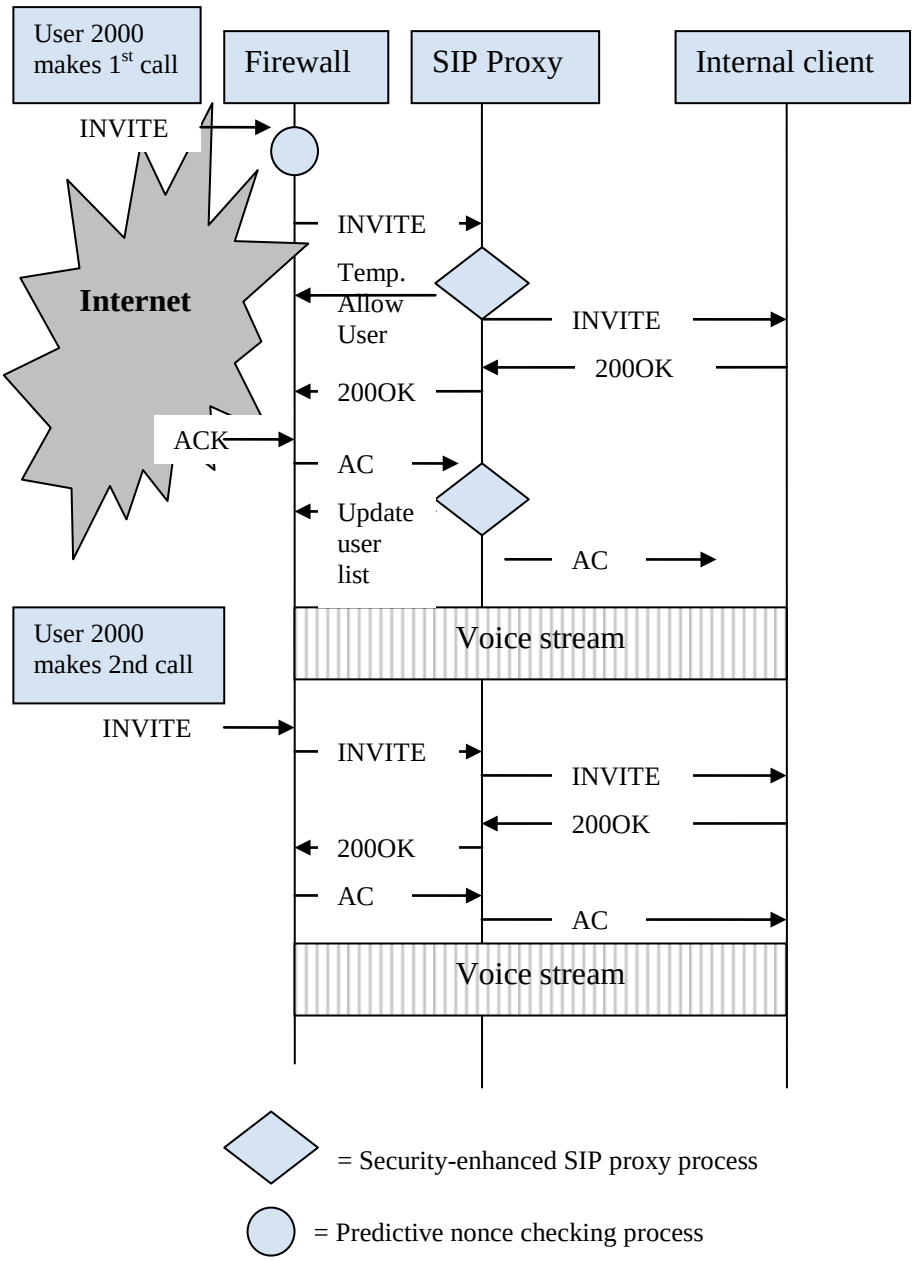


Figure 31: the improved security-enhanced SIP system

In ISESS, the firewall will drop all SIP request messages except INVITE and REGISTER from an unknown user until such time as the user has been successfully authenticated using an INVITE or REGISTER. For known users, the process in this system is the same as

that in SESS, i.e. requests will be passed to the proxy server directly through iptables. Entries on the frequent user list, normal user list and temporary user list will be removed on expiry.

For unknown users, however, the system will do the following:

- When the request arrives at the firewall, it will check to see if it is an INVITE or REGISTER. If so, the firewall will perform the predictive nonce checking authentication as explained in section 4.4.1. If the packet is another SIP request, the firewall will drop it. This ensures no spoofed SIP requests are passed through the firewall.
- When the INVITE message arrives at the SIP proxy, since the packet has already passed the authentication checks on the firewall, the SIP proxy does not have to challenge it again.

Once the SIP Proxy receives a message, it does the following:

- The SIP proxy will firstly check if the source IP address is already on the *userlist* or *frequent userlist*, if so, the server will process it in the same way as in SESS.
- If it is not on any lists, the SIP proxy will temporarily add the source IP address to its *temporary user list* for 30 seconds (the reason for choosing 30 seconds is that it is a typical waiting time before an unanswered phone is directed to voicemail). Within the 30 seconds, if the callee answers the phone, the OK and ACK messages will be exchanged.
- When the server receives the ACK, the user will be moved from the *temporary user list* to the *userlist* that has been specified in section 4.3. After the user is added to the userlist, a KASP message will be sent to the firewall to update the changes on the *userlist*, and the process is the same as SESS.

Using this process, all legitimate users will continue to receive the same performance as that in SESS, while all forms of SIP request flooding will be blocked by the firewall.

4.6.2 ISESS analysis

ISESS is an advance on SESS by eliminating other types of SIP request flooding attacks while maintaining good QoS for legitimate users. This is for the following reasons:

Only INVITE and REGISTER requests are allowed from unknown users: By using this filtering rule at the firewall, other types of SIP requests can be eliminated.

The use of a temporary user list ensures that unknown legitimate users can become authenticated without any difficulties.

The firewall queuing, userlist and frequent userlist update mechanisms are the same as in SESS, which maintains a good QoS for legitimate users during flood attacks.

Chapter 6 of this document will explain the implementation of SESS and ISESS followed by experiments to examine their performances.

Chapter 5: Implementation and test results

This chapter explains the implementation procedure of the proposed SESS and ISESS. After implementing the systems, a series of experiments are conducted to verify the performance of each system under SIP DoS attacks. The performance metrics used in the experiments are mainly call setup delays for different types of users (unknown, normal and frequent users) and CPU usages on both SIP proxy server and the firewall during an attack. The experiment results should show that when using SESS and ISESS, during an INVITE or REGISTER flooding attack the call setup delays for normal users and frequent users should not be affected by the attack, while the call setup delay for unknown users should be longer than usual, as it has to go through layer-7 authentication process. ACK flooding attacks will be conducted to examine whether SESS is vulnerable to other type of SIP request flooding attacks. We can expect that during an ACK flooding attack, the call setup delay for all users under SESS will increase dramatically. When ISESS is configured, during other types of SIP flooding attacks, we can expect that no discernable effect will be observed. This chapter details the way the experiments are conducted, followed by some systematic analysis of the experimental results.

5.1 Implementation of SESS

5.1.1 Security-enhanced SIP proxy server

5.1.1.1 Choice of implementation platform

The enhanced SIP proxy server is implemented in Java using JAIN SLEE [92]. The reason for choosing JAIN SLEE [92] is that it is designed for telecommunication's low latency and high throughput environments (10-20 calls per second per CPU; ~10 events per call;

<200ms RTT) [93]. It also enables easy integration of new capabilities using a high level language - Java. Jain SLEE is one of the most advanced network service environments available for network service development.

5.1.1.2 Implementation details

When implementing security-enhanced SIP proxy server, the SIP service component on JAIN SLEE server, called SIP Service Building Block is modified. Each incoming ACK message is logged and the IP address of the sender of this message is stored on one of the user list, depending on the time it made the last phone call. Hashtables are used to store the legitimate user IP addresses. There are two static hashtables created: *frequentuserlist,* and *userlist*. User source IP addresses are used as hashtable keys, and user objects are stored as hashtable values.

The main Java objects created are as follows:

- A user object is created, which contains three attributes: a user source IP, a timer object, and a current time when the user request is handled.
- Timer objects are in charge of expiring the entry on its list. When the entry expires, the timer object will call a timer task object, which will perform a user removal action. If the user is removed from a *frequentuserlist*, it will be added to the *userlist*. If a user is removed from the *userlist*, the user will be considered to be unknown when they next make a phone call. The *frequentuserlist* has a shorter life cycle than the *userlist*. In the testbed environment, the *frequetuserlist* is set to expire after 10 minutes, and the *userlist* to expire after 15 minutes.

When a user completes the INVITE three-way handshake, the proxy server will carry out the enhanced security process as described previously.

Figure 32 shows the pseudo code of the userlist class.

```
If (user is already on the frequentuserlist) {
        Update timer;
}or if (user is on the userlist) {
        If (Timedifference < frequentuserlist expire time) {
                Userlist.remove (user);
                Frequentuserlist.put(user);
                reset userTimer to frequentuserTImer;
                notify firewall about the change;
        }or else{
                Reset userTimer to userTimer;
        }
}or else {
        Userlist.put (user);
        Notify firewall of the addition of user;
        Set timer to userTimer;
}
```

Figure 32: the pseudo code of the userlist class.

Figure 33 illustrates the pseudo code of the remove user process.

```
Remove user (userIP, which list it is on, timer) {
        If (the user is on the frequentuserlist) {
                Frequentuserlist.remove(userIP) ;
                Userlist.put(userIP, user);
                Notify firewall;
        )or if (the user is on the userlist) {
                Userlist.remove(userIP);
                Notify firewall;
        }
}
```

Figure 33: the remove user process.

If the user is removed from the *frequentuserlist*, it will be added to the *userlist*, and a demotion notification will be sent to the firewall. If the user is removed from the *userlist*, notification of removal will be sent to the firewall.

When the firewall receives a notification, it will map the notification to an iptables rule. The next section will explain how this is done.

5.1.2 Implementation of the security enhanced firewall

The security enhanced firewall is implemented using a standard linux iptables firewall. Iptables rule sets are a true layer-three process with no application layer processing required, and this ensures optimal performance of the system. This section explains how the firewall is implemented.

5.1.2.1 DNAT and regular housekeeping

The first requirement for the firewall is to enable destination network address translation (DNAT), and do regular SIP firewall setup. DNAT is used to prevent exposure of the SIP proxy private address. All requests that are received at the firewall's external interface on port 5060 will be forwarded to the SIP proxy. This is done by using the iptables DNAT rule set, eg:

```
iptables -t nat -A PREROUTING -p tcp -i eth0 -d 10.0.0.1 --dport 5060 -j DNAT --to 192.168.1.62:5060
```

Then, the stateless SIP firewall is set up. After DNAT, all incoming SIP messages will be sent to the FORWARD chain. Thus, to ensure all SIP packets are passed to the stateless predictive nonce checking, we need to specify a firewall rule set as follows:

```
iptables –A FORWARD –p udp –i eth0 –d 192.168.1.62 –-dport 5060 –j queue
```

5.1.2.2 Firewall rule set update daemon

Updates to the rule set are achieved using a separate C daemon. When the daemon starts up, it should flush the current iptables rule sets and re-establish the predefined rules as above.

This program then listens on the port 1117 of the firewall's internal interface (IP address: 192.168.1.1) where the userlist updates from the proxy server are received. When an update packet is received, the daemon reads the payload and converts the KASP message to an iptables rule. The KASP conversion is as described in the following table (table 4):

Table 4: KASP conversion details

Bytes 6-8	Action	Rule
+nu	**INSERT**	**ACCEPT FORWARD**
-nu	**DELETE**	**ACCEPT FORWARD**
+fu	**INSERT**	**ACCEPT FORWARD**
	APPEND	**PREROUTING** with **TOS** to **Minimize Delay**.
-fu	**DELETE**	**ACCEPT FORWARD**
	DELETE	**PREROUTING** with **TOS** to **Minimize Delay**.

To execute the converted iptables command, use a method called *system (command string)*.

5.2 Implementation of ISESS

SIP proxy server modification:

When an INVITE message is received, the SIP server will first check to see if its source IP address is already on the *userlist* or *frequent userlist*. If so, do nothing. Otherwise, the server needs to add the IP address to a temporary user list *(tempUserList*), and set the

timer to expire in 30 seconds, and then send the updated information to the firewall. For simplicity, the updated information will be the KASP *nu+* message, so we do not have to modify the firewall for a new type of message. When the timer expires, the user will be removed from the list, and a KASP *nu-* will be sent. These procedures are performed by a new method called *checkUser()*. Figure 34 shows the pseudo code of *checkUser()*.

```
If (user is not already on the frequentuserlist && it is not on the
userlist) {

        tempUserList.put (user);
        Notify firewall about the addition of user;
        Set timer to 30 seconds;
```

Figure 34: Pseudo code for checkUser()

In SESS, when an ACK is received from a new user it will be added to the *userlist*. In ISESS, an ACK can only be added to a *userlist* if it is already on the *tempUserList*. Figure 35 shows the pseudo code for this process.

```
If (user is already on the frequentuserlist) {
        Update timer;
}or if (user is on the userlist) {
        If (Timedifference < frequentuserlist expire time) {
                Userlist.remove (user);
                Frequentuserlist.put(user);
                reset userTimer to frequentuserTImer;
                notify firewall about the change;
        }or else{
                Reset userTimer to uerTimer;
        }
}or else {
        Userlist.put (user);
        Notify firewall about the addition of user;
        Set timer to userTimer;
}
```

Figure 35: Pseudo code for adding user, when ACK is received.

Advanced firewall modification:

In the predictive nonce checking module, ISESS will drop any packet other than an INVITE and REGISTER. The rest of the program is the same as SESS. The iptables rule set update daemon is used to update new iptables rule sets for legitimate users.

5.3 Test results

A number of trials were carried out to verify the effectiveness of the proposed solutions, with particular emphasis on improvements in the performance of the application-layer stateless firewall. It is important to note that the test results are applicable to our testbed. The results may differ if tested in other systems. In the following experiments, for simplicity SIP INVITE flood is used in most of the system performance tests. This is because SIP INVITE flood is considered to be a more harmful SIP attack traffic, as it requires more

processing power to process. As stated in section 1.1, the objective of this project is to find a solution that is able to stop most SIP flood traffic, meanwhile maintaining a good QoS for legitimate users. In order to verify how the proposed solutions fit the criteria, the following factors are measured:

Call setup delays when the system is under SIP flood attacks are measured, which is an important criterion to measure the QoS of a telephony system.

Call setup timeout percentage: is used to measure the availabilities of a system. As mentioned earlier, the general availability objectives of PSTN network are to achieve 99.999%, which means the call setup timeout has to be less than 0.001%.

CPU usages on the firewall and SIP proxy server when the system is under flood attack. This helps to illustrate how moving authentication from the SIP proxy to the firewall can help to improve the performance of the system.

Other SIP requests floods are used to verify how the proposed systems mitigate flooding of other types of SIP requests. For example, ACK, and CANCEL flood.

It is important to note that the way the length of iptables rulesets affects the performance of the system is measured, however its effect is not significant enough, and will not be discussed in this document. Stress tests have been conducted with a large amount of flood traffic and the test results show SESS and ISESS can still provide good performance under this attack. Since this is not a critical examination factor, the details of this experiment will not be discussed.

5.3.1 Call setup delays for new users, normal users and frequent users

This section studies the call setup delays for different users (new user, normal user and frequent user) under various levels of security (no security, stateless firewall, SESS and ISESS.

During the experiment, a constant INVITE flood of 1000 packets per chunk, with one second delay was used (this specific chunk size and inter-chunk delay were arrived at based on intensive trial runs with varying sizes and delays). During these trials, we found that if the attack rate is too low, there would be little impact on the performance of the system. If the attack rate is too high, most of the attack traffic will be lost due to the network congestion. With 1000 packets per chunk delay of one second, we are able to block all incoming calls, and a packet loss rate of 9.2 %., is used to stress the system. In order to test how the length of the two stage list may affect the performance of the system, 100 users were manually added to the *frequentuserlist*, and 100 users to the *userlist*. The iptables rule sets were also updated to reflect these list entries.

When the system has no security activated, or when the system is under the protection of the basic application-layer stateless firewall, there is no difference in processing each of the types of users and so there will be no difference in call setup delay. Thus, for both of these configurations, a single set of call setup delay data will suffice. However, with SESS and ISESS configurations, the call setup delay for new users, normal users and frequent users should differ and so the call setup delay for the three types of users is measured separately.

Figure 36 shows the average call setup delay for different users under various security levels.

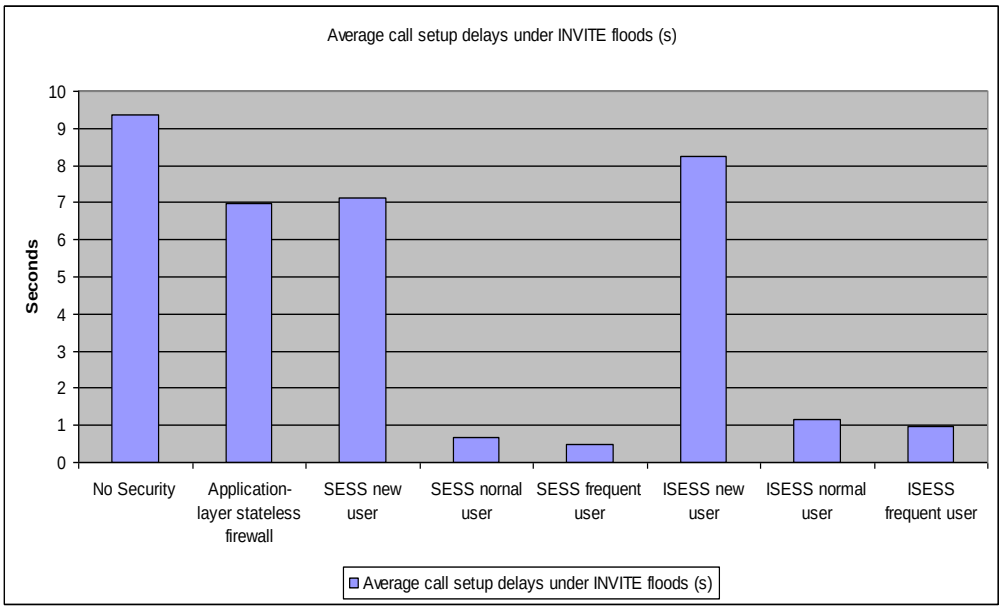


Figure 36: Average call setup delays for different users.

The average call setup delay for users under no security is 9.39 (this calculation does not include the call setup timeouts) seconds.

With the application-layer firewall, the average call setup delay for users under application-layer predictive nonce checking firewall is measured at 7.06 seconds, which is still quite high.

With the security enhanced SIP system (SESS), there were no timeouts. The average call setup delay for new users was 7.14 seconds which is consistent with the application layer stateless firewall, as expected. However, the call setup delay was measured at 0.675 seconds for normal users and 0.487 seconds for frequent users – a substantial improvement with little impact on new user call setup.

With ISESS, the average call setup delay for new users is 8.26 seconds, and 1.14 seconds for normal users. For frequent users, the average call setup delay during an INVITE flood is 0.97 seconds. Compared to SESS, the call setup delay is a bit longer, however this can be disregarded.

5.3.2 Call setup timeout percentages during flooding attacks.

In this experiment, the CPU usages on both firewall and SIP proxy server are measured during an INVITE flood attack. The CPU usage on the two components will be compared under each level of security as for the previous set of experiments.

Figure 37 illustrates the experimental results.

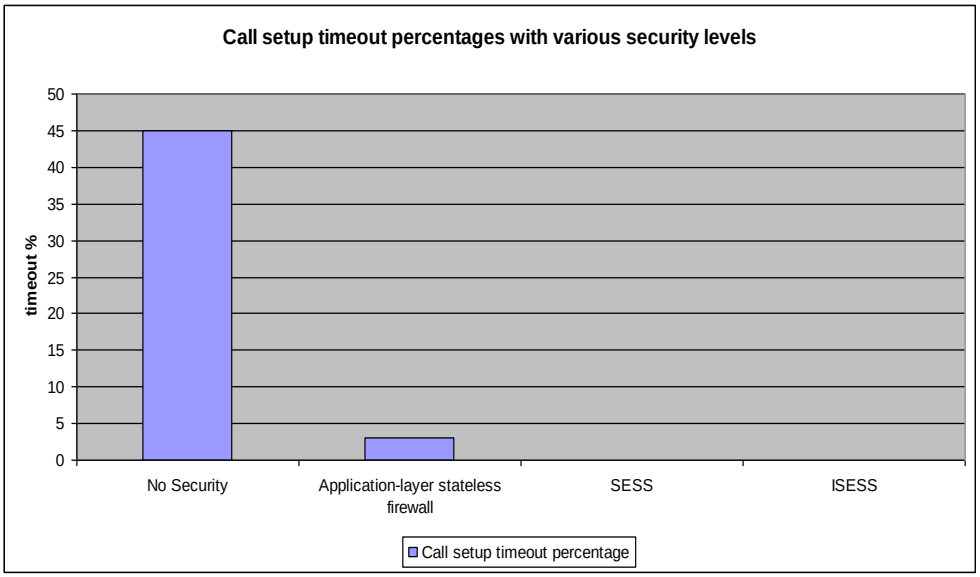


Figure 37: Average call setup timeouts under various security levels

As we can see from figure 37, the average call setup timeout during an INVITE flood, under no security is 14 times higher than that under application-layer stateless firewall. With SESS and ISESS, there are no setup timeouts, which improves the performance of this system significantly.

5.3.3 CPU usages on the firewall and SIP proxy server during an attack

This experiment measures the CPU usages on both firewall and SIP proxy server during an INVITE flood when the system is under various security levels. Figure 38 illustrates the experimental results.

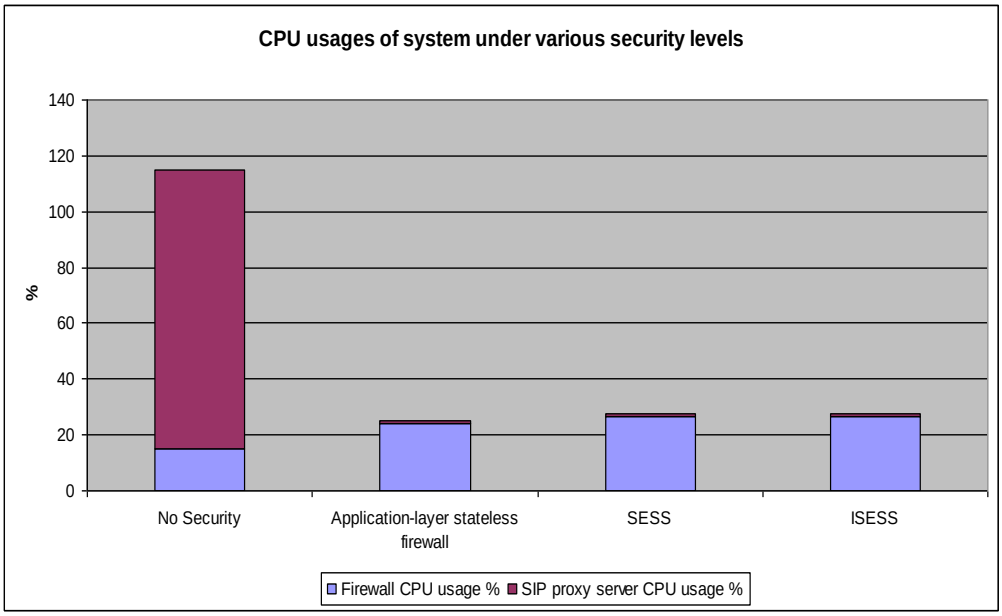


Figure 38: Comparison of CPU usages on firewall and SIP proxy server under various security levels.

As can be seen, when there is no security deployed in the system the flooding attack can easily overload the SIP proxy server with spoofed requests (the server is almost at 100% CPU) and result in the high call setup timeout rate seen in the previous experiments. The average CPU usage at the firewall is only 15%, which indicates the firewall is not fully utilized.

When an application-layer stateless firewall is used the firewall will have to carry out much more processing than it does with its basic iptables forwarding (no security) rules. During the attack the CPU usage on the firewall is only increased to 24% (a 9% increase) and it is still able to process other incoming requests. The CPU usage on the

SIP proxy server is 1%. This is because only legitimate requests are allowed to pass through the firewall, the SIP proxy server only processes a limited number of requests, and thus even though the system is under an attack the CPU usage on the proxy server should not be affected by the attack traffic. Thus with authentication at the firewall, load balance is achieved and the SIP proxy server does not become overloaded by the attack. Further, there was no call setup timeout under the stateless application-layer predictive nonce checking system.

When using SESS the average CPU usage on the firewall increases to 26.5% due to spoofed requests being authenticated. The CPU usage on the proxy server is similar to that seen in the application layer stateless firewall configuration despite the fact that the server is in charge of updating user lists – i.e. the list update is very efficient and does not happen too often.

When ISESS is configured, the average CPU usage on the firewall is the same as that in SESS, because the methods to process spoofed SIP requests are identical on both systems. The CPU usage on the SIP proxy is 1% which shows that no attack packet has passed through the firewall.

5.3.4 ACK flood on SESS and ISESS

SESS is designed to stop spoofed INVITE and REGISTER flooding requests, as these can be challenged by digest authentication. However, flooding attacks based on other spoofed SIP requests such as the ACK packet pass through the firewall without any authentication and so are not countered by SESS. ACK packets are processed by the SESS proxy to find out the source IP of a legitimate user. Thus, a large amount of spoofed ACK can exhaust the resource on the SESS SIP proxy server. In order to verify this hypothesis, an ACK flooder tool

was developed based on iFlood, to send out ACK requests. The tool allows the 'attacker' to specify how many packets are to be sent, the rate of attacking traffic, and the range of spoofed source IP addresses. Additionally, it allows the attacker to specify the user ID to use. This can be useful, as some SIP proxy servers are restricted to processing requests with a SIP URL-formatted source address.

ISESS is considered to be a mitigation technique that is able to stop any type of SIP request floods. In order to verify this hypothesis, a series of experiments are conducted to verify the performance of the systems under ACK floods.

As with the earlier experiments, the ACK flooding experiment was carried out with spoofed traffic at 1000 packets per chunk with one second delays.

Call setup delays during ACK floods for SESS and ISESS:

Figure 39 compares the average call setup delays for frequent users during ACK floods under SESS and ISESS.

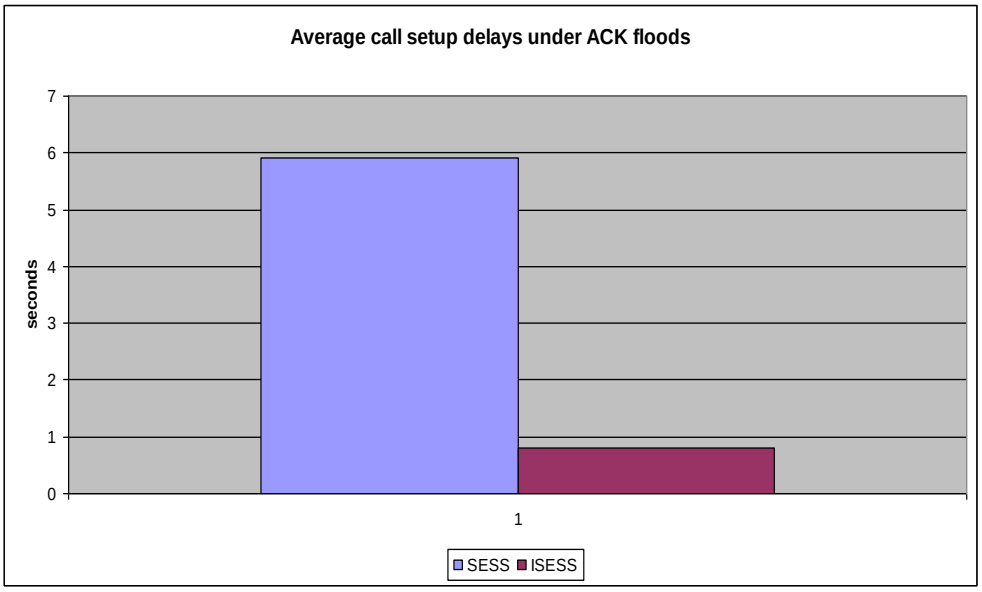


Figure 39: Comparison of average call setup delays under ACK floods with SESS and ISESS

From figure 39 we can see that the average call setup delay in SESS is 5.90 seconds during ACK floods (this calculation excludes the call setup timeouts). The reason is that if the spoofed ACK are passed through to the SESS proxy server, it will record the source IP addresses as legitimate and eventually crash due to lack of memory space. As expected, the ACK flood does affect the performance of SESS.

When ISESS is configured, the average call setup delay for a frequent user when the system is under an ACK flood attack was measured at 0.815 seconds. This reduces the call setup delay in SESS by 86%. Also, none of the attack packets was captured on the SIP proxy server, which means the firewall still has a very high (in our experiment this is 100%) success rate in blocking spoofed ACK requests.

This 0.815 second call setup delay is partially caused by network congestion. The network congestion happens when a chunk of attack traffic is sent, and can be identified in the above diagram as the peaks of call setup delays.

This test result also shows that during an INVITE flood, the average call setup delay is slightly greater (0.155 seconds) than with an ACK flood as the firewall has to send back digest authentication messages to the spoofed addresses. In order to verify this assumption, we measured the call setup delays for a frequent user, when the ISESS is under INVITE flooding attack and compared them with those under ACK flooding attack.

Call setup timeout during ACK floods for SESS and ISESS

Figure 40 compares the average call setup timeout percentages during an ACK flood attack for SESS and ISESS.

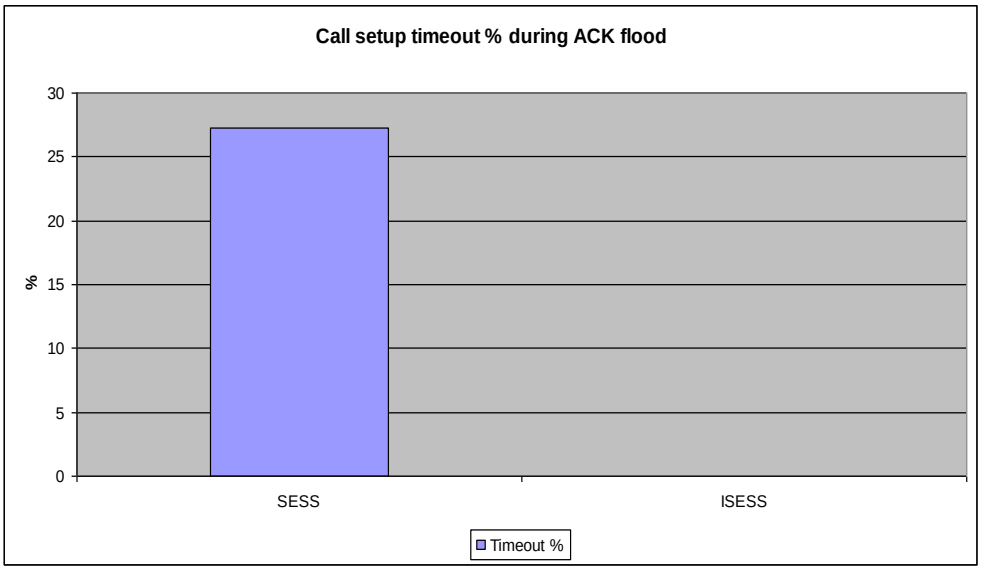


Figure 40: Comparison of call setup timeout percentages for SESS and ISESS during ACK floods

When the system is configured with SESS, during an ACK flooding attack the average call setup timeout for users is 27.3%. However, with ISESS it is reduced to 0%. This is because in SESS, all spoofed ACK requests are passed through the firewall and processed by the SIP proxy server which has to store all spoofed source IP addresses in its memory, and finally the server will crash due to lack of memory space. This is the reason for high call setup timeouts in the previous experiment. Figure 41 shows the log message when the SIP proxy server crashes.

The server error log clearly illustrates the reason for the 500 Server Internal Error is lack of memory. Thus, the ACK flood attack can exhaust the memory resource on the SESS server and cause a denial of service attack.

This experiment verifies that ISESS is able to block other SIP request floods, apart from INVITE and REGISTER. In the following section, we furthered the experiments using PROTOS test suite, which

can generate spoofed CANCEL messages to see how ISESS mitigates this type of attacks.

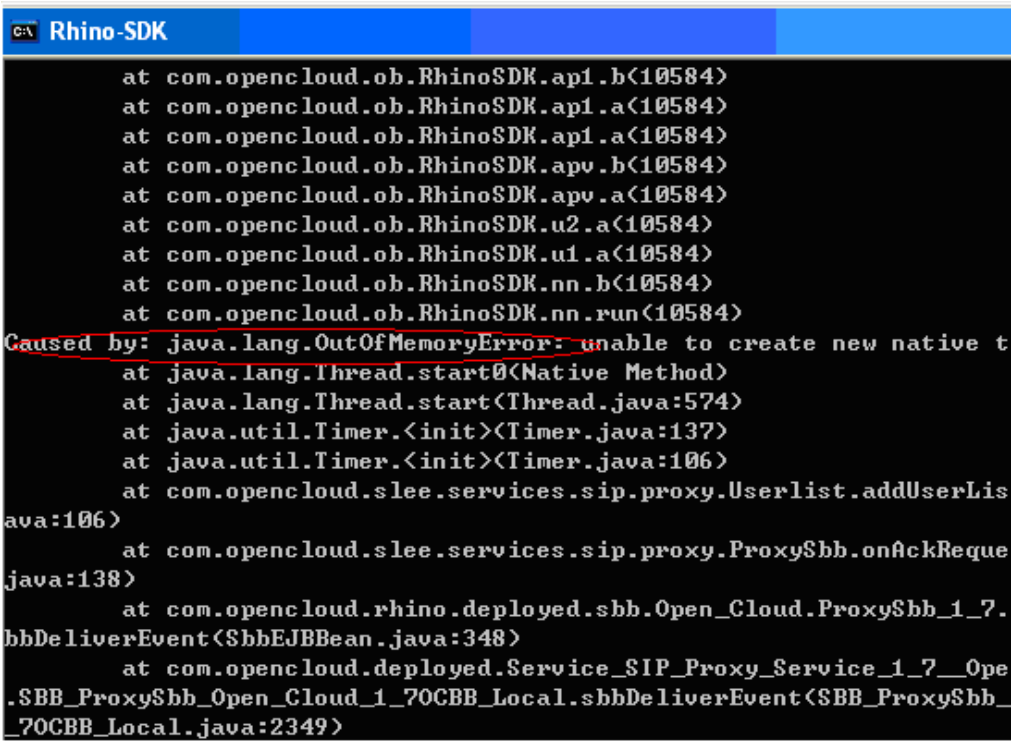


Figure 41: Server crash log message

5.3.5 Other SIP request floods against ISESS

The PROTOS program [94]was developed at the University of Oulu in Finland as an inexpensive way to test protocol implementations for security vulnerabilities. The PROTO test suite c07-sip is used to test the robustness of SIP protocol. This test suite can be used to generate session teardown streams, where a sequence of spoofed INVITE, CANCEL and ACK requests are sent to the proxy server. The attacker can specify the content of each header field (to form either malformed message attack or overflow attack) to cause a denial of service problem on the SIP proxy server. The test results from the PROTO group have demonstrated that SIP is vulnerable to a number of malformed attacks.

In our experiment, we use PRORO c07-sip as an attacker generating session teardown packets (INVITE, CANCEL and ACK messages with corresponding CallIDs). The reason to choose teardown attack is because this attack generates all the most common SIP requests. Thus, we can test how well ISESS defeats the spoofed SIP request attack. Figure 42 [94] illustrates the session teardown process.

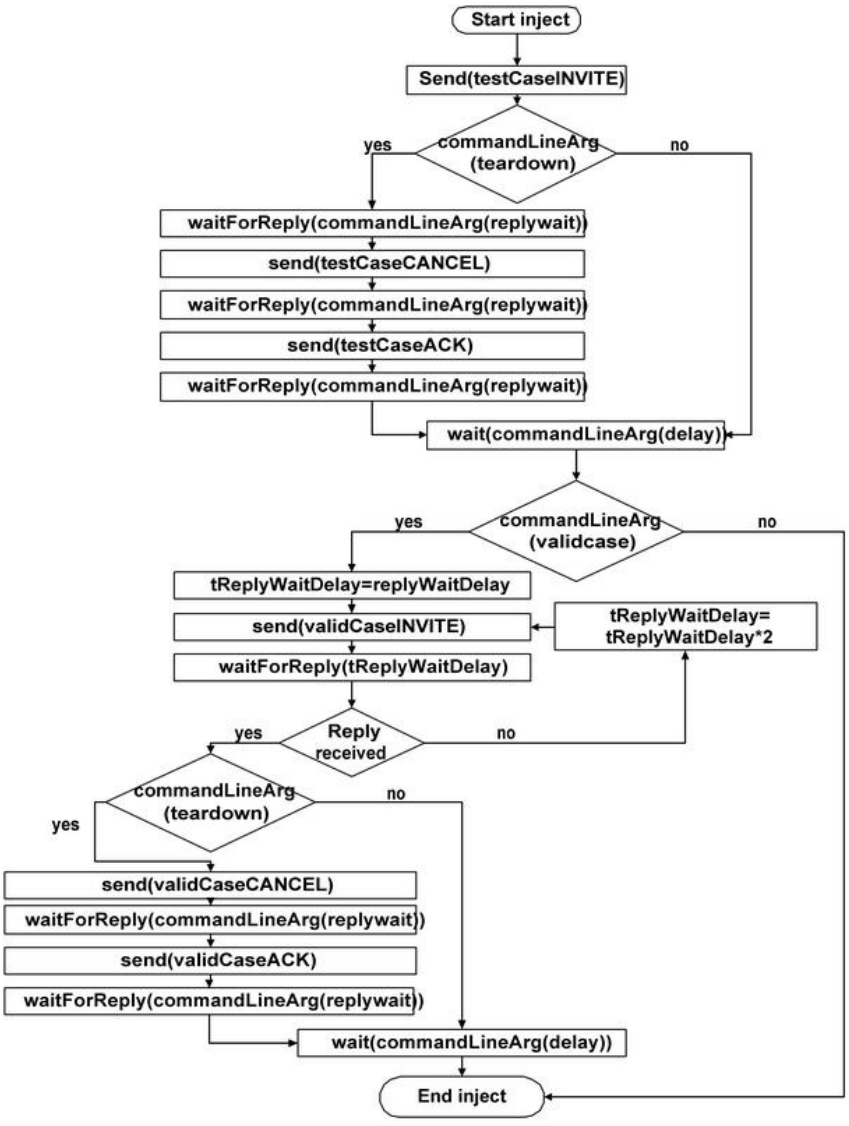


Figure 42: PROTO teardown process

In our test bed, when a teardown attack is launched, the packets sent by PROTO are illustrated in Figure 43.

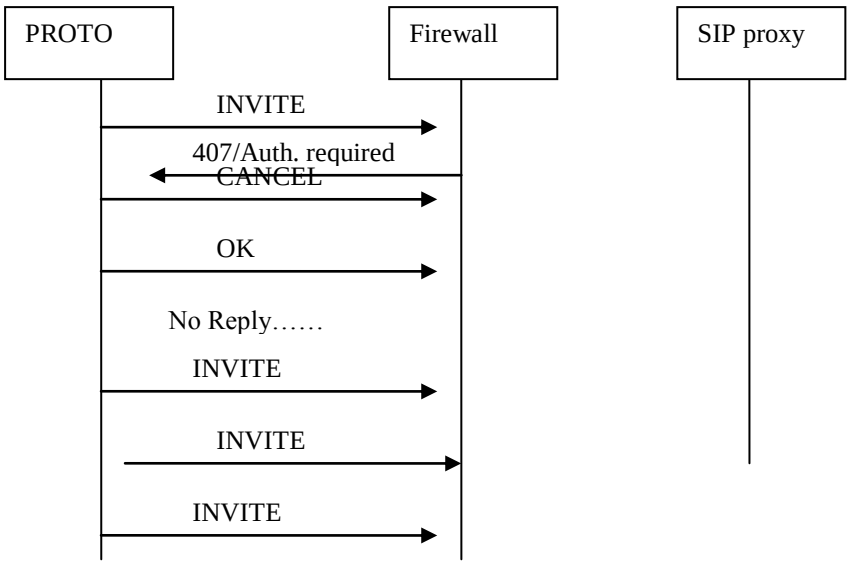


Figure 43: PROTO SIP teardown attack packet flow diagram

The PROTO first sends an INVITE request, followed by CANCEL and OK requests. As specified in RFC 3261, Section 9.1 states: The Request-URI, CallID, To, the numeric part of CSeq, and From header fields in the CANCEL request MUST be identical to those in the request being cancelled, including tags. Since there is no expected reply for the first INVITE request after sending a sequence of SIP session teardown requests the PROTO will repeatedly send out INVITE requests.

We used wireshark to monitor the number of attack packets passed through the firewall. From our experiment, we observed that no attack packet had passed through the ISESS firewall. This is because PROTO c07-sip does not have a SIP digest authentication protocol stack in its application, so the authentication challenge cannot be processed by the test suite. Thus INVITE messages cannot pass through the firewall authentication. Without passing the firewall authentication, other SIP requests will be dropped by the firewall.

This experiment proves that ISESS is able to defeat all types of SIP request flooding attacks.

5.4 Analysis and Conclusion

The trials clearly indicate that both SESS and ISESS have an extremely high (experimental results how a 100%) success rate in blocking SIP INVITE and REGISTER floods, while maintaining a good QoS for legitimate users. This is because when attackers use spoofed source IP addresses, they will never receive the digest challenge message and so cannot calculate the nonce value. All spoofed SIP requests will fail the firewall authentication, and get dropped. Since SESS and ISESS place the requests from known users in a different queue from other requests, no matter how severe the flooding attack is, as long as there is enough bandwidth to transfer the requests, the call setup delays for legitimate users are not affected by the attack.

The call setup delay experiment proves that SESS and ISESS can improve the call setup delay by 94.8% (=0.487s/9.39s) and 89.6% (= 0.97/9.39s) compared to no security. This occurs because requests from legitimate users are directly passed through the iptables rules to the proxy server and do not need application level processing and authentication, so the latency due to the firewall process is negligible.

Experimental results also show that for frequent users SESS and ISESS provide a 93.7% (=0.487s/7.06s) and 86.2% (= 0.97s/7.06s) improvement of call setup delay compared to an application-layer stateless firewall when the system is under INVITE flooding attacks. This is because in an application-layer stateless firewall, requests from legitimate users and spoofed INVITE requests will be treated in the

same way by the firewall - they will all be passed from the network interface to an application-layer nonce checking process. Since in the stateless firewall scenario, all requests have to be authenticated with predictive nonce authentication, and each request has to be processed twice, which is very processing intensive, and time consuming, thus, the call setup delay is increased. Plus, as there is only a single FIFO queue to pass the packets to the application layer, the time for packets waiting in the queue increases. Thus, when the system is under flooding attack the call setup delay for legitimate users will increase. However, by using SESS, known legitimate users' requests are passed to the SIP proxy directly, so do not need to wait in the queue to be authenticated by the firewall.

The average call setup time for a frequent user in ISESS during INVITE flood is twice as long as that in SESS (0.97 seconds). This is caused by the extra check on the received INVITE request on the proxy. Since this increase in delay is not significant (0.49 seconds), it can be disregarded.

The average call setup delay difference for a frequent user and a normal user is very similar. This is because the call setup delay is just caused by the prioritization at the firewall, and this process is the same in both SESS and ISESS.

The call setup delay for new users in ISESS is significantly longer (about 1 second) than that in SESS and the application-layer firewall. This is because in ISESS when a new user tries to place a call, an extra KASP message is processed to add that user to the "temporary userlist" on both firewall and SIP proxy. This process increases the call setup delay for new users.

The call setup delay differences between normal users and frequent users in both SESS and ISESS are very similar (0.2 and 0.17

seconds). It is important to note that this time difference has little to do with the iptables rule matching delay because the ACCEPT iptables rules for the normal user and frequent user are inserted at the top of the iptables rule sets. This delay difference is caused by the differentiated Type of Service (ToS) settings for frequent users, where frequent users are signed with "Minimize-Delay". Thus, SIP requests from frequent users are passed to the SIP proxy server with the highest priority, and requests from normal users are not prioritised. This feature should provide the same QoS for known users even when the system is under severe flooding attacks because the spoofed requests are sent to the INPUT queue to be authenticated by the firewall, while requests from known hosts are sent to the FORWARD queue and will be forwarded regardless of how congested the INPUT queue is.

The time for known user lists expiry will affect the call setup delays for all users to an extent. The shorter the expiry time, the longer the average call setup time. This is partially because when an entry in the lists expires, the SIP proxy has to fork a thread to perform the action of removing the user from the corresponding list, and send a KASP message to the firewall. This process would require some CPU power, and if this happens too often, the performance of the SIP system can degrade. The other reason is that if a user is removed from the known user lists too soon, then the probability that he is treated as a new user is higher. At the extreme, if the user expiry time is set to 0 second, this system setup delay will be similar to that in the application-layer stateless firewall, as the user has to go through the firewall authentication every time he makes a call. This would increase the average call setup time for this user.

Contrarily, the performance of the system would also degrade if the user list expiry time is set too high, as this could result in a very

long legitimate user list and contention for RAM in the SIP Proxy server. Thus, it is important to select an optimal userlist size.

From the above experiments we can see that ISESS is able to block any types of spoofed SIP request, even though the average call setup delay for users is slightly longer than with SESS. Since the increase in call setup delay can hardly be noticed by human beings, ISESS is a preferable solution for SIP flood attacks. It is worth noting that the ISESS system cannot compensate for a heavily congested network. When the network is congested, most of the packets will be randomly dropped by the intermediate routers. In an extreme situation, legitimate requests may never be able to reach the firewall.

Chapter 6: Conclusion and future work

6.1 Other considerations

6.1.1 SIP Botnet attacks

A Botnet attack [95] is caused when a number of Internet computers are subverted without their owners' knowledge in order to forward transmissions (including spam or viruses) to other computers on the Internet. Botnets are used in the majority of DDoS attacks. According to the VoipSA blog [96], SIP botnets can be created even though there is no evidence of a clear and present threat. Nassar et al [97] have developed a proof-of-concept SIP Internet Relay Chat (IRC) botnet, which is able to send successive INVITE requests, and spoofed REGISTER requests, as well as scan urls of legitimate users. As shown in figure 44, ISESS will be able to counter this type of botnet.

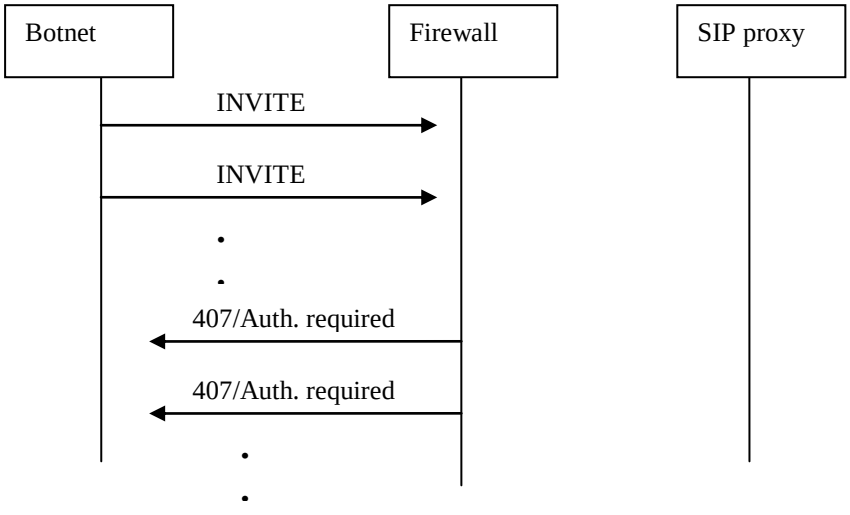


Figure 44, ISESS counters simple botnet attacks

In this simple botnet attack scenario, the attacking bots are not attached to any SIP protocol stacks or SIP clients, and the bots are very similar to any other SIP flooding tools. The bots can either use the real

source IP addresses or spoofed addresses. In this case, since no SIP protocol stack is installed on the attacking machines, the authentication challenge messages cannot be processed, and no legal re-INVITE message can be generated. Thus, no attack request can pass through the firewall, when ISESS is used.

Even though there is no evidence of advanced SIP botnet, there is a potential for attack tools to be created which fully understand the SIP protocol, and can be hooked up with real SIP clients, and so generate legal SIP requests and respond to any authentication requests. As shown in figure 45, ISESS will not be able to counter this type of attack because it considers all SIP requests that accomplish a three-way handshake to be legitimate.

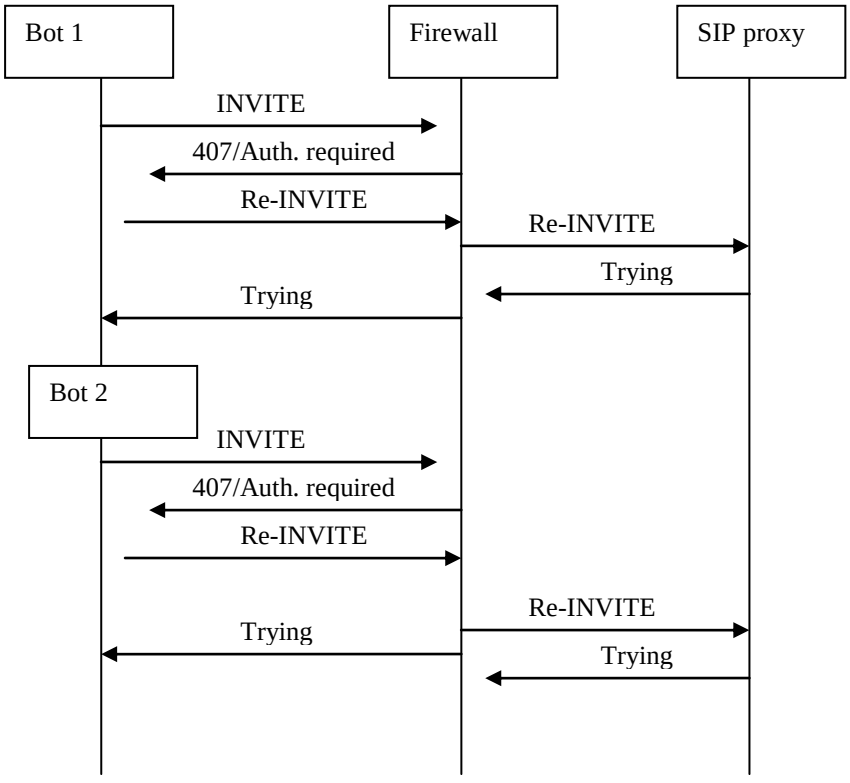


Figure 45: Advanced botnet on ISESS

Since the bots are all legal SIP clients, they will initiate SIP sessions in exactly the same manner as legitimate SIP clients so ISESS cannot distinguish this attack from normal session initiations. When all advanced botnet traffic passes through the firewall, and reaches the SIP proxy server, the server will be totally occupied by these requests, and this could result in a DoS attack on the system. If the attack traffic is targeting an existing client, this can also cause DoS on that particular client.

6.1.2 ISESS in the real-world scenario

6.1.2.1 Global view

In the real-world scenario, SIP proxies are normally interconnected with each other across the globe to enable international VoIP sessions, as shown in figure 46. The communication between two hosts usually involves the request forward among different SIP proxies.

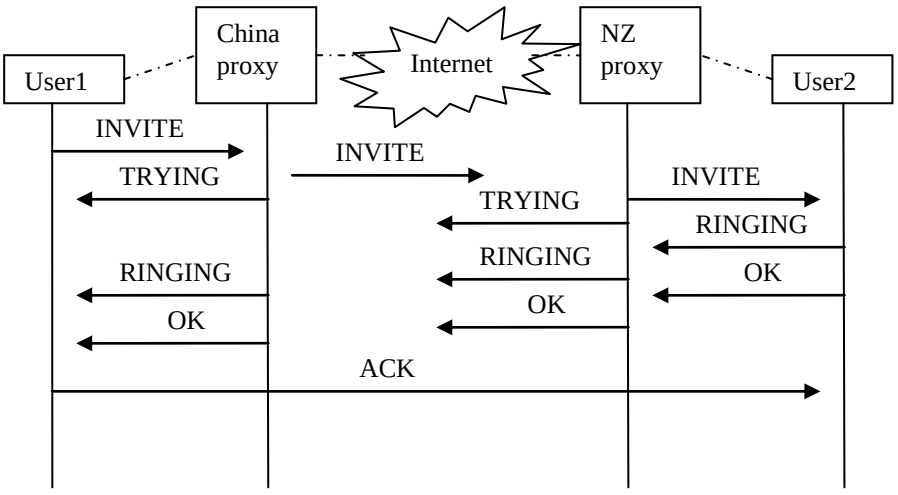


Figure 46: SIP call setup in a global scenario

Even though the network infrastructure is different, the call setup process in a global scenario is very similar to that in a local setup. The

SIP proxy servers are still vulnerable to flooding attacks originating either from their own internal networks, or from the Internet. The way to launch an attack is the same as that launched on our testbed.

6.1.2.2 Session Border Controller

In some scenarios, ISPs may deploy a session border controller (SBC) [98] to exert control over signalling and media traffic. An SBC normally acts as a back-to-back user agent (B2BUA) and is essentially an application layer gateway and proxy situated between the participants of a session. It receives a request and processes it as a user agent server, then it determines how the request should be answered, and acts as a user agent client to generate requests. The caller would recognize this B2BUA as the called party. This helps to hide the internal address of a particular SIP user agent.

Security features on SBCs differ from vendor to vendor. Acme packet SBC [99] provides SIP flooding protection by limiting the bandwidth of "unknown hosts". Figure 47 explains the DoS mitigation mechanism on ACME packet SD.

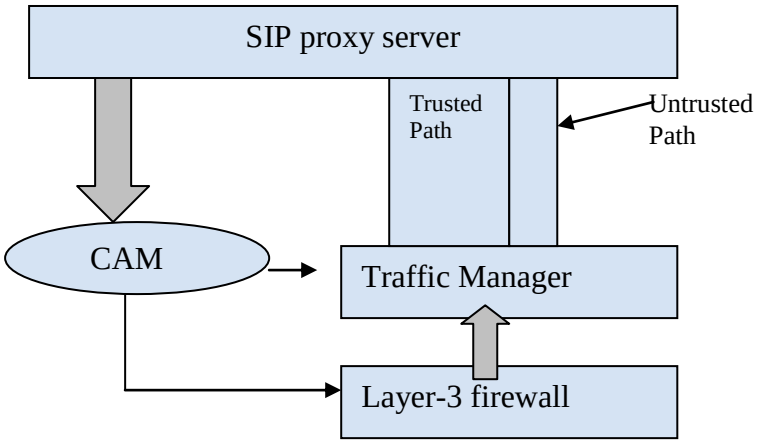


Figure 47: DoS mitigation on ACME packet SD.

An access control list (ACL) is used to eliminate access from "malicious hosts". Since the firewall and SIP proxy are built on the same host, the ACL is stored at the content addressable memory (CAM) for easy modification and sharing between the SIP proxy and the layer-3 firewall. When a packet comes in, if it is from a blocked host, the layer-3 firewall will drop it. Otherwise, it will be passed to a traffic manager, which is responsible for categorizing the packet as either "trusted" or "untrusted". The untrusted packets will be passed to a limited path, and sent to the SIP proxy server. The server will update the ACL by either promoting a host as "trusted", or demoting a host as "blocked" based on the activity of that host.

The ACL-based access on one hand could help reduce the impact of a DDoS attack from a known attacker. However, since SIP attackers normally use spoofed IP addresses, it is likely that this approach would not be very effective. Additionally, since this approach is host-based, i.e. IP address-based, there is a risk that the attacker would spoof an IP addresses of a legitimate user and so cause a denial of service to the legitimate users.

SBCs are beyond the scope of this project however, it provides ideas on whether to integrate the ISESS system into a single host. This would be a useful area for future research.

6.2 Conclusion and future work

As the dominant VoIP session initiation and management protocol, SIP is susceptible to various attacks, especially flood-based DoS attacks. Some of the existing commercial firewalls, for example: AR450 from AlliedTelesis have SIP anti-flooding mechanisms to protect a SIP proxy server from DoS attacks. However experimental results have shown that the anti-flood mechanisms on these firewalls

may be defeated. Existing work has identified some countermeasures, however they all have various limitations and some of them are not practical to implement. This project has identified and trialled a number of countermeasure designs.

From the experiments carried out during the course of this research, a Security Enhanced SIP System (SESS) was developed. SESS consists of an enhanced SIP-aware firewall and an enhanced SIP proxy server which communicates using a protocol called KASP. SESS extends and synthesises existing research to create an advanced SIP security capability which was demonstrated to be very effective against a SIP INVITE or REGISTER flooding attack. However, further testing using other flooding attacks demonstrated that SIP ACK flooding traffic could pass through the firewall and achieve its target of denial of service.

Subsequently, an Improved Security Enhanced SIP System (ISESS) was developed by enhancing SESS further in order to counter a wide range of SIP flooding attacks. Using ISESS, all SIP flooding traffic is blocked at the firewall, no matter what type of attack packet is used.

ISESS counters the flooding attack effectively and avoids timeout due to SIP proxy overloads. The performance of ISESS is good for known users even under a severe flooding attack. The average call setup delay for a frequent user in ISESS is just under a second, and for a normal user just over a second. Unknown user setup is about 7 seconds.

While ISESS will handle any form of flooding packet, and some forms of botnet attack, a fully SIP-aware botnet attack using real addresses would be indistinguishable to ISESS from legitimate traffic.

Further research is needed to address this form of future botnet capability.

References:

[1] N. Banerjee, S. Saklikar, and S. Saha, "Anti-vamming trust enforcement in peer-to-peer VoIP networks." pp. 201-206.

[2] Y. Soupionis, S. Dritsas, and D. Gritzalis, *An Adaptive Policy-Based Approach to SPIT Management*: Springer, 2008.

[3] D. Geneiatakis, G. Kambourakis, T. Dagiuklas *et al.*, "SIP Security Mechanisms: A state-of-the-art review," *Proc. 5th International Network Conference (INC)*, pp. 147–155.

[4] H. Schulzrinne, and J. Rosenberg, "A Comparison of SIP and H. 323 for Internet Telephony," *Proc. International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, pp. 83–86.

[5] D. Endler, "VoIP hacking exposed," *p.121,147,167,369,389,429,487,505*, McGraw-Hill, 2007.

[6] E. T. Aire, B. T. Maharaj, and L. P. Linde, "Implementation considerations in a SIP based secure voice over IP network," *AFRICON, 2004. 7th AFRICON Conference in Africa*, vol. 1, 2004.

[7] E. C. Cha, H. K. Choi, and S. J. Cho, "Evaluation of Security Protocols for the Session Initiation Protocol," *Computer Communications and Networks, 2007. ICCCN 2007. Proceedings of 16th International Conference on*, pp. 611-616, 2007.

[8] Q. Qiu, "Study of Digest Authentication for Session Initiation Protocol (SIP)," December, 2003.

[9] S. Salsano, L. Veltri, and D. Papalilo, "SIP security issues: the SIP authentication procedure and its processing load," *Network, IEEE*, vol. 16, no. 6, pp. 38-44, 2002.

[10] S. McGann, and D. C. Sicker, "An Analysis of Security Threats and Tools in SIP-Based VoIP Systems," *Proceedings of the 2 ndWorkshop on Securing Voice over IP, Cyber Security Alliance*, 2005.

[11] V. D. Gligor, "A NOTE ON THE DENIAL-OF-SERVICE PROBLEM," *Proceedings of the 1983 Symposium on Security and Privacy, April 25-27, 1983, Oakland, California*, 1984.

[12] P. Hunter, "VOIP the latest security concern: DoS attack the greatest threat," *Network Security*, vol. 2002, no. 11, pp. 5-7, 2002.

[13] shawnmer, "Hackers send thousands of fake calls to deaf people," *VOIPSA, http://voipsa.org/blog/2008/03/20/hackers-*

send-thousands-of-fake-calls-to-deaf-people/, Mar. 2008, Accessed 1 Oct 2008.

[14] D. Sass, "Voice over IP Security Planning, Threats and Recommendations, http://www.infosecwriters.com/text_resources/pdf/VOIP_DSass.pdf," 2006,, Accessed Aug 2007.

[15] S. J. Templeton, and K. E. Levitt, "Detecting spoofed packets," *DARPA Information Survivability Conference and Exposition, 2003. Proceedings*, vol. 1, 2003.

[16] T. Chiba, T. Katoh, B. B. Bista *et al.*, "DoS Packet Filter Using DNS Information," *Proceedings of the 20th International Conference on Advanced Information Networking and Applications-Volume 1 (AINA'06)-Volume 01*, pp. 116-131, 2006.

[17] B. E. Fredrik Thernelius, "SIP Firewall Solution," *http://www.softarmor.com/wgdb/docs/draft-thernelius-sip-firewall-solution-00.txt*, Accessed Sep. 2007.

[18] J. Kim, S. Yoon, Y. Won *et al.*, "VoIP Secure Communication Protocol satisfying Backward Compatibility," *Systems and Networks Communications, 2007. ICSNC 2007. Second International Conference on*, pp. 43-43, 2007.

[19] A. Yaar, A. Perrig, and D. Song, "SIFF: a stateless Internet flow filter to mitigate DDoS flooding attacks," *Security and Privacy, 2004. Proceedings. 2004 IEEE Symposium on*, pp. 130-143, 2004.

[20] CCS-WG, "Quality of Service (QoS) Standard for Telephone Services," *http://www.ofta.gov.hk/en/ad-comm/tsac/cc-paper/ccs2005p11.pdf*, 2005, Accessed Feb.2008.

[21] D. Endler, and M. Collier, *Hacking Exposed VoIP: Voice Over IP Security Secrets & Solutions (Hacking Exposed)*: McGraw-Hill Osborne Media, 2006.

[22] I. Rec, "H. 323, Packet based Multimedia Communications Systems, http://www.itu.int/rec/T-REC-H.323/en/," *Telecommunication Standardization Sector of ITU*, Accessed Jul 2007, 2003.

[23] M. Handley, H. Schulzrinne, E. Schooler *et al.*, "SIP: Session Initiation Protocol," vol. 2543, 1999.

[24] H. Schulzrinne, S. Casner, R. Frederick *et al.*, "RFC1889: RTP: A Transport Protocol for Real-Time Applications," *Internet RFCs*, 1996.

[25] RFC768, "UDP, User Datagram Protocol, IETF Standard," August, 1980.

[26] I. Rec, "H. 245," *Control protocol for multimedia communication*, *http://www.itu.int/rec/T-REC-H.245/en/*, 1998, Accessed Jul. 2007.

[27] I. Rec, “H. 225,” *Call signalling protocols and media stream packetization for packet-based multimedia communication systems* , *http://www.itu.int/rec/T-REC-H.225.0/en/*, 1998, Accessed Jul 2007.

[28] M. Shore, *H. 323 and Firewalls: Problem Statement and Solution Framework*, *https://datatracker.ietf.org/drafts/draft-shore-h323-firewalls/*, Internet Draft, Accessed Aug 2007.

[29] T. Berners-Lee, L. Masinter, and M. McCahill, "Uniform Resource Locators (URL)," RFC 1738, CERN, Xerox Corporation, University of Minnesota, December 1994 (http://ds. internic. net/rfc/rfc1738. txt), 1994.

[30] T. Berners-Lee, R. Fielding, and H. Frystyk, "RFC2616 - Hypertext Transfer Protocol -- HTTP/1.1," May, 1996.

[31] I. Packetizer, “H.323 versus SIP: A Comparison,” *http://www.packetizer.com/ipmc/h323_vs_sip/*, Accessed May/2007.

[32] R. Rivest, “RFC1321: The MD5 Message-Digest Algorithm,” *Internet RFCs*, 1992.

[33] M. Zandi, M. V. Martin, and P. C. K. Hung, “Overview of security issues of VOIP,” *IASTED European Conference on Proceedings of the IASTED European Conference: internet and multimedia systems and applications table of contents*, pp. 254-259, 2007.

[34] P. C. K. Hung, and M. V. Martin, “Through the looking glass: Security issues in VoIP applications,” *Proceedings of the IADIS International Conference on Applied Computing*.

[35] P. A. Version, “Eavesdropping an IP Telephony Call.”

[36] US-CERT, “Understanding Denial-of-Service Attacks,” *http://www.us-cert.gov/cas/tips/ST04-015.html*, 2008.

[37] L. Garber, “Denial-of-Service Attacks Rip the Internet, ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=839316,” 2000, Accessed Sep 2007.

[38] H. Wang, D. Zhang, and K. G. Shin, “Detecting SYN flooding attacks,” *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 3, 2002.

[39] C. C. Center, “CERT Advisory CA-1996-21 TCP SYN Flooding and IP Spoofing Attacks,” *Internet: http://www. cert. org/advisories/CA-1996-21. html, September*, 1996, Accessed Sep 2007.

[40] J. Farrell, "IP Fragmentation Attacks on Checkpoint Firewalls, www.giac.org/certified_professionals/practicals/gsec/0589.php ", April, 2001. Accessed Dec 2007.

[41] D. Forte, “Fragmentation Attacks: Protection Tools and Techniques Called “true preliminaries to denial-of-service”,

IpFrags are a tough nut to crack for some firewalls and intrusion detection systems," *Network Security*, vol. 2001, no. 12, pp. 12-13, 2001.

[42] M. Kenney, "Ping of Death, insecure.org/sploits/ping-o-death.html," *Insecure. org*, 1996, Accessed Sep. 2007.

[43] J. Albers, B. Hahn, S. McGann *et al.*, "An analysis of security threats and tools in SIP-based VoIP Systems," *Retrieved April*, vol. 4, pp. 2006, 2005.

[44] L. Juranic, *Using fuzzing to detect security vulnerabilities*, Technischer Bericht INFIGO-TD-01-04-2006, Infigio Information Security, Zagreb, Kroatien, April 2006.

[45] R. Baumann, S. Cavin, and S. Schmid, "Voice Over IP-Security and SPIT," *Swiss Army, FU Br*, vol. 41.

[46] J. Quittek, S. Niccolini, S. Tartarelli *et al.*, "Detecting SPIT Calls by Checking Human Communication Patterns," *Communications, 2007. ICC'07. IEEE International Conference on*, pp. 1979-1984, 2007.

[47] S. Niccolini, "SPIT prevention: state of the art and research challenges," *Network Laboratories, NEC Europe, Germany*.

[48] M. Hansen, M. Hansen, J. M 鰈 ler *et al.*, "Developing a Legally Compliant Reachability Management System as a Countermeasure against SPIT," *Third annual security workshop (VSW?6)*.

[49] B. Sterman, D. Schwartz, and E. Katz, "DETECTION OF SPIT IN VOIP CALLS," WO Patent WO/2006/126,202, 2006.

[50] A. Patrizio, "Vishing Joins Phishing as Security Threat," *Internet News, http://www. internetnews. com/security/article. php/3619086, accessed July*, vol. 11, 2006.

[51] T. Chan, S. Sengodan, N. R. Center *et al.*, "On applying SIP security to networked appliances," *Networked Appliances, 2002. Gaithersburg. Proceedings. 2002 IEEE 4th International Workshop on*, pp. 31-40, 2002.

[52] J. McCarron, "A Brief Overview of VoIP Security."

[53] J. Zar, "VoIP Security and Privacy Threat Taxonomy," *Public Release*, vol. 1, pp. 24, 2005.

[54] P. Ferguson, and D. Senie, "RFC2267: Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing," 1998.

[55] R. K. C. Chang, "Defending against flooding-based distributed denial-of-service attacks: a tutorial," *Communications Magazine, IEEE*, vol. 40, no. 10, pp. 42-51, 2002.

[56] H. Wang, C. Jin, and K. G. Shin, "Defense against spoofed IP traffic using hop-count filtering," *IEEE/ACM Transactions on Networking (TON)*, vol. 15, no. 1, pp. 40-53, 2007.

[57] D. Kashiwa, E. Y. Chen, and H. Fuji, "Active shaping: a countermeasure against DDoS attacks," *Universal Multiservice Networks, 2002. ECUMN 2002. 2nd European Conference on*, pp. 171-179, 2002.

[58] A. Bremler-Barr, R. Halachmi-Bekel, I. C. Herzliya *et al.*, "Unregister Attacks in SIP," *Secure Network Protocols, 2006. 2nd IEEE Workshop on*, pp. 32-37, 2006.

[59] E. Nuwere, and M. Varpiola, "The Art of SIP Fuzzing and Vulnerabilities found in SIP," *BlackHat Conference, Las Vegas, NV, July*, 2005.

[60] A. Hussain, J. Heidemann, and C. Papadopoulos, "A framework for classifying denial of service attacks," *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pp. 99-110, 2003.

[61] M. Collier, "Basic Vulnerability Issues for SIP Security, http://download.securelogix.com/library/SIP_Security030105.pdf," *Research Report*, 2005, Accessed Sep. 2007.

[62] K. Egevang, and P. Francis, "The IP Network Address Translator (NAT)," RFC 1631, May 1994, 1994.

[63] D. Endler, and M. Collier, "Hacking Exposed VoIP: Voice Over IP Security Secrets & Solutions (Hacking Exposed)," *http://hackingvoipexposed.com/sec_tools.html*, pp. 395-398, 2006.

[64] Xianglin Deng, and C.-W. Lee, "Security of VoIP-SIP flooding and its Mitigation," *New Zealand Computer Science Research Student Conference 08'*, 2008.

[65] H. Aljifri, "IP Traceback: A New Denial-of-Service Deterrent?," 2003.

[66] J. Ioannidis, and S. M. Bellovin, "Implementing Pushback: Router-Based Defense Against DDoS Attacks," *Proceedings of Network and Distributed System Security Symposium*, vol. 2, 2002.

[67] M. Kim, and K. Chae, "Detection and Identification Mechanism against Spoofed Traffic Using Distributed Agents," *Time*, vol. 26, no. 31, pp. 36.

[68] J. Li, J. Mirkovic, M. Wang *et al.*, "SAVE: source address validity enforcement protocol," *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 3, 2002.

[69] H. L. Flanagan, "Egress filtering–keeping the Internet safe from your systems," *Online at http://rr. sans. org/sysadmin/egress. php, April*, 2001, Accessed Dec 2007.

[70] T. Peng, C. Leckie, and K. Ramamohanarao, "Protection from distributed denial of service attacks using history-based IP

filtering," *Communications, 2003. ICC'03. IEEE International Conference on,* vol. 1, 2003.

[71] H. Aljifri, "IP traceback: a new denial-of-service deterrent?," *Security & Privacy Magazine, IEEE,* vol. 1, no. 3, pp. 24-31, 2003.

[72] M. T. Goodrich, "Efficient packet marking for large-scale IP traceback," *Proceedings of the 9th ACM conference on Computer and communications security*, pp. 117-126, 2002.

[73] S. M. Bellovin, "ICMP Traceback Messages," 2003.

[74] M. Sung, and J. Xu, "IP Traceback-based Intelligent Packet Filtering: A Novel Technique for Defending Against Internet DDoS Attacks," *IEEE Transactions on Parallel and Distributed Systems,* vol. 14, no. 9, pp. 861-872, 2003.

[75] D. X. Song, and A. Perrig, "Advanced and authenticated marking schemes for IP traceback," *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE,* vol. 2, 2001.

[76] H. Sengar, H. Wang, D. Wijesekera *et al.*, "Fast Detection of Denial of Service Attacks on IP Telephony," *Proceedings of the 14th IEEE International Workshop on Quality of Service (IWQoS 2006)*, 2006.

[77] S. Fowler, and S. Zeadally, "Defending against Distributed Denial of Service (DDoS) Attacks with Queue Traffic Differentiation over Micro-MPLS-based Wireless Networks," *Network,* vol. 10, pp. 11, 2006.

[78] B. Reynolds, and D. Ghosal, "Secure IP telephony using multi-layered protection," *NDSS Symposium, San Diego, CA*, 2003.

[79] Y. You, M. Zulkernine, and A. Haque, "Detecting Flooding-Based DDoS Attacks," *Communications, 2007. ICC'07. IEEE International Conference on*, pp. 1229-1234, 2007.

[80] H. Sengar, D. Wijesekera, H. Wang *et al.*, "VoIP Intrusion Detection Through Interacting Protocol State Machines," *Dependable Systems and Networks, 2006. DSN 2006. International Conference on*, pp. 393-402, 2006.

[81] E. Y. Chen, "Detecting DoS attacks on SIP systems," *VoIP Management and Security, 2006. 1st IEEE Workshop on*, pp. 53-58, 2006.

[82] D. Geneiatakis, and C. Lambrinoudakis, "An ontology description for SIP security flaws," *Computer Communications,* vol. 30, no. 6, pp. 1367-1374, 2007.

[83] T. Peng, C. Leckie, and K. Ramamohanarao, "Survey of network-based defense mechanisms countering the DoS and DDoS problems," *ACM Computing Surveys (CSUR),* vol. 39, no. 1, 2007.

[84] J. Rosenberg, *Request Header Integrity in SIP and HTTP Digest Using Predictive Nonces*, expired Internet draft, work in progress, IETF, June 2001.

[85] M. Ohta, "Overload Protection in a SIP Signaling Network," *Internet Surveillance and Protection, 2006. ICISP'06. International Conference on*, pp. 11, 2006.

[86] N. Aschenbruck, M. Frank, P. Martini *et al.*, "Present and Future Challenges Concerning DoS-attacks against PSAPs in VoIP Networks," *Proceedings of the Fourth IEEE International Workshop on Information Assurance, April*, pp. 13-14, 2006.

[87] S. Ehlert, G. Zhang, D. Geneiatakis *et al.*, "Two layer Denial of Service prevention on SIP VoIP infrastructures," *Computer Communications*, 2008.

[88] J. Jung, B. Krishnamurthy, and M. Rabinovich, "Flash crowds and denial of service attacks: characterization and implications for CDNs and web sites," *Proceedings of the eleventh international conference on World Wide Web*, pp. 293-304, 2002.

[89] R. Sailer, and M. Kabatnik, "History based distributed filtering-a tagging approach to network-level access control," *COMPUTER SECURITY APPLICATIONS*, pp. 11-15, 2000.

[90] S. D. D'Souza, and D. Vinokurov, "Queuing methods for mitigation of packet spoofing, http://www.google.co.nz/patents?hl=en&lr=&vid=USPATAPP10712103&id=fsmfAAAAEBAJ&oi=fnd&dq=queuing+method+for+mitigation+of+packet+spoofing," *US Patent App*, 2004, Accessed Feb. 2008.

[91] R. Droms, "Dynamic Host Configuration Protocol," RFC 2131, March 1997, 1997.

[92] S. B. Lim, and D. Ferry, "JAIN SLEE 1.0 Specification, final release," *Sun Microsystems, Inc. and Open Cloud Limited March*, 2004.

[93] M. Mareztke, "JAIN SLEE technology overview," *http://www.maretzke.de/pub/lectures/jslee_overview_2005/JSLEE_Overview_2005.pdf*, 2005.

[94] U. o. Oulu, "PROTOS - Security Testing of Protocol Implementations," *http://www.ee.oulu.fi/research/ouspg/protos/*, Accessed July 2008.

[95] R. Puri, "Bots & botnet: An overview," *SANS Institute'03*, 2003.

[96] VOIPSA, *http://voipsa.org/blog/2007/05/07/ready-or-not-here-come-the-irc-controlled-sipvoip-attack-bots/*, 2008.

[97] Mohamed Nassar, Radu State, and O. Festor, "VoIP-IRC bot," *http://www.loria.fr/~nassar/readme.html*, 2008.

[98] G. Camarillo, "Functionality of Existing Session Border Controller (SBC), http://tools.ietf.org/html/draft-camarillo-sipping-sbc-funcs-00," *IETF Draft, February14*, vol. 4, 2005, Accessed Mar 2008.

[99] A. Packet, "Acme Packet Net-Net Session Border Controllers," *Product information*, 2004.