

# SpamWall



## Syndicate

NC Tahir Abbas

PC Naveed Khalid

PC Mudassar Lateef

## Project Supervisor

Col. Naveed Sarfaraz Khattak

## **DEDICATION**

In the name of Allah, the Most Merciful, the Most Beneficent

To our parents and teachers without whose unflinching support and unstinting cooperation, a work of this magnitude would not have been possible

# ACKNOWLEDGMENTS

First of all we would like to thank Allah Almighty. Whatever, we have achieved; we owe it to Him, in totality.

We are really grateful to our parents, family and well-wishers for their admirable support not during the course of the project, but also throughout our lives for without them, all this would have never been possible.

We gratefully recognize the continuous supervision and motivation provided to us by our Project Supervisor, Col. Naveed Sarfaraz khattak (HOD CS Dept.). We are highly thankful to Lec. Aisha Khalid Khan, Lec. Tabinda Waheed, Lec Bilal Basheer (OIC MIS Cell) and System Administrator (MIS Cell) Mr. Jaffar Hussain, who had been guiding and supporting us throughout our course and research work. Their knowledge, guidance and training enabled us to carry out this research work.

## Table of Contents

<b>Chapter 1</b> .....	<b>1</b>
1Introduction.....	1
1.1 Introduction .....	1
1.2 Problem Statement .....	1
1.3 Proposed Solution .....	1
1.4 Motivation .....	2
1.5 Project Scope Statement.....	2
1.6 Research Domain .....	3
1.7 Organization of Report.....	3
1.8 Summary .....	4
<b>Chapter 2</b> .....	<b>5</b>
Research and Study.....	5
2.1 Introduction .....	5
2.2 Spam.....	5
2.2.1 Types of Spam .....	5
2.2.2 Analysis of Different Types of Spam .....	11
2.3 Socket Programming.....	11
2.3.1 JDBC.....	12
2.3.2 Javamail .....	12

2.3.3	JEC (Java Exchange Connector).....	13
2.3.4	EWSJ (Exchange Web Services Java).....	13
2.4	Mail protocols .....	13
2.4.1	IMAP Protocol.....	14
2.4.2	POP3 Protocol.....	14
2.4.3	SMTP Protocol.....	14
2.4.4	Relay Protocols .....	16
2.5	Configuration of Exchange Server.....	18
2.6	Spam Filtering Algorithms.....	18
2.6.1	Spam Firewall.....	18
2.6.2	Using Blacklists .....	19
2.6.3	Using White-Lists .....	19
2.6.4	Spam Database.....	20
2.6.5	Bayesian Filtering.....	20
2.6.6	Heuristic Filtering .....	22
2.6.7	Challenge Response Spam Filtering .....	23
2.7	Analysis of Filtering Techniques .....	24
2.8	Structure of Firewall.....	24
2.8.1	Firewall Types .....	24
2.9	Java Mail Server (JES).....	25

2.10	Summary .....	25
<b>Chapter 3</b>	<b>.....</b>	<b>26</b>
	Requirements Specifications.....	26
3.1	Introduction .....	26
3.1.1	Purpose.....	26
3.1.2	Conventions .....	26
3.2	Overall Description .....	27
3.2.1	Product Perspective.....	27
3.2.2	Product Features.....	27
3.2.3	Top Level Data Flow Diagram .....	28
3.2.4	User Classes and Characteristics .....	29
3.2.5	Operating Environment.....	29
3.2.6	Design and Implementation Constraints.....	29
3.2.7	User Documentation .....	30
3.3	System Features.....	30
3.3.1	Enable SpamWall.....	30
3.3.2	Disable SpamWall .....	30
3.3.3	Filter Mail .....	31
3.3.4	Scan Attachments.....	31
3.3.5	System Log .....	31

3.3.6	Forward Mail .....	32
3.4	Other Nonfunctional Requirements .....	32
3.4.1	Performance Requirements.....	32
3.4.2	Security Requirements .....	32
3.4.3	Software Quality Attributes .....	33
3.5	Summary .....	33
<b>Chapter 4</b>	<b>.....</b>	<b>34</b>
Software Design.....		34
4.1	Introduction .....	34
4.2	Design Consideration.....	34
4.2.1	General Constraints.....	34
4.3	Goals.....	35
4.4	Development Methods .....	35
4.5	Architectural Strategies .....	36
4.6	Main Components (Modules) of SpamWall: .....	36
4.6.1	Receiving Module:.....	37
4.6.2	Filtering Module: .....	38
4.6.3	Forwarding Module: .....	38
4.7	Class Diagram .....	<b>Error! Bookmark not defined.</b>
4.8	Use Case.....	41

4.9	Sequence Diagram.....	44
4.10	Dataflow Diagram.....	45
4.11	Summary .....	46
<b>Chapter 5</b>	<b>.....</b>	<b>37</b>
Implementation	.....	47
5.1	Introduction .....	47
5.2	Implementation Language.....	47
5.3	Implementation of JES.....	47
5.4	Implementation of Bayesian.....	48
5.5	Implementation of Blacklist Filtering Algorithm.....	49
5.6	Implementation of White-List Filtering Algorithm .....	49
5.7	Implementation of link Checker Algorithm.....	49
5.8	Implementation of Email Scanner Algorithm .....	49
5.9	Summary .....	53
<b>Chapter 6</b>	<b>.....</b>	<b>40</b>
Testing & Analysis	.....	54
6.1	Introduction .....	54
6.2	Validation and Verification.....	54



6.3	Unit Testing.....	55
6.4	Integration Testing .....	55
6.5	Black Box Testing.....	55
6.6	System Testing .....	56
6.7	Summary .....	56
<b>Chapter 7</b>	<b>.....</b>	<b>43</b>
Conclusion & Future Work.....		57
7.1	Introduction .....	57
7.2	Enhancement in Filtering Algorithms .....	57
7.3	Enhancement in the Compatibility.....	57
7.4	Enhancement in the Functionality.....	58
7.5	Increase in Actions .....	58
7.6	Interaction of End User .....	58
7.7	Conclusion.....	58
7.8	Summary .....	59
<b>Bibliography</b>	<b>.....</b>	<b>Error! Bookmark not defined.</b>

## **Chapter 1**

# **Introduction**

### **1.1 Introduction**

This chapter describes the problem statement, proposed solution, motivation for this project and the overall organization of this document. Project named “SpamWall” is an Email filtering application. Its primary task is to filter unwanted email called spam from the original email known as ham.

### **1.2 Problem Statement**

Spam is unsolicited e-mail which is sent in massive quantities to Internet email users. Spam is undesirable because of its disadvantages i.e., loss of resources, loss of money, wastage of time, user Inconvenience and many more. Spam is almost impossible to stop completely, but it can be reduced to a less aggravating level through spam filters.

Many anti-spam solutions/tools are available but they are too expensive and not easily affordable. This is the main reason that MCS has shifted its mail server to third party (Google) for anti-spam services.

### **1.3 Proposed Solution**

To design and develop an anti-spam software application/tool (SpamWall) to filter spam from original email using best email filtering algorithms. Algorithms used in SpamWall are: Blacklist Algorithm, White list Algorithm, Bayesian filtering Algorithm, Hyperlinked based Algorithm.

Attachments contained within email can also be a security threat as it can have malwares i.e. Spyware, Adware, Viruses etc. To cater this problem SpamWall is linked with existing anti-virus software to scan for suspicious attachments. Proposed solution is described in figure 1.1.



Figure 1.1 Architectural model

## 1.4 Motivation

Information Security (email security) is an emerging field in Pakistan and it is being used for variety of purposes in a variety of applications. Project would not only be a learning task but also a practical benefit for the MIS Cell (MCS). In this respect this project is very important, as mentioned before currently MCS doesn't have this facility (anti-spam filter), obviously causing a lot of problems for email users at MCS and therefore shifted its mail server to third party (Google) for anti-spam services. The project is requirement and an offer by MIS Cell MCS.

## 1.5 Project Scope Statement

The project is aimed at developing a software application to provide anti-spam service for an existing mail server particularly for MIMS. Purpose of this project (SpamWall) is to filter spam from email using filtering algorithms i.e. blacklisting/white listing, Bayesian filtering algorithm and hyperlinked based algorithm. This software application is also linked with existing anti-virus software to scan for the suspicious attachments in the email. Given an email spam filter will check for: Text messages (classification of text), Empty messages, Suspicious web

links, Linking with existing antivirus to scan the attachments for malwares i.e. Spyware, Adware and Viruses etc.

## 1.6 Research Domain

The domain SpamWall is dealing is the information security as in figure 1.2. Further in information security, there are different sub domains which include Network layer security, Operating system security, Database security and Application layer security. The application layer security further contains sub domains such as P2P application security, Email security and Web application security. Area of focus and research is the Email security.

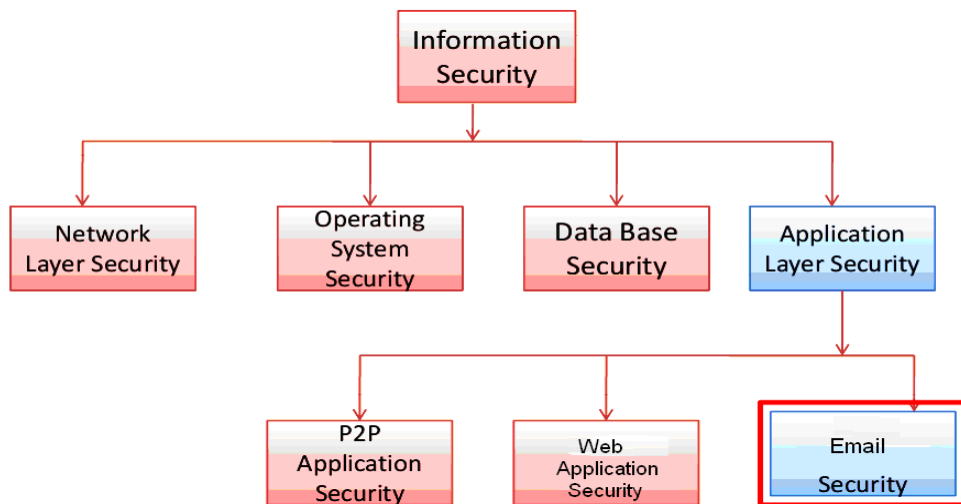


Figure 1.2: Area of Research

## 1.7 Organization of Report

This project report has been divided into eight chapters. Chapter 1 gives an introduction to the technology used and project statement and motivation behind under taking the project. Chapter two gives the literature review in detail along with the appropriate references. Chapter 3 is based on the detailed analysis of system requirements. Chapter 4 gives the brief system

description along with system constraints. Chapter 5 describes the system design and architecture and explains the way project is organized. Chapter 6 describes the system development with all the details of the system functions and explains the way they have been developed. Testing and validation of the system is explained in chapter 7 and finally chapter 8 describes all the achievements of the project and the forums where it has been presented also including the future work that can be done in the project.

## **1.8 Summary**

This chapter described the details about the Email security and the importance of the project as a requirement of MIS Cell. It also described the problem at hand, its solution and domain of work.

## Chapter 2

# Research and Study

### 2.1 Introduction

Few areas needed to be studied in order to get understanding of some basic knowledge required for developing SpamWall. Topics which were studied are: Spam and Its types, Socket programming (APIs e.g. JEC, ESWJI, Javamail), Mail protocols, Configuration of exchange server, Spam filtering algorithms, Structure of Firewall, Java Mail Server (JES).

### 2.2 Spam

E-mail spam, known as unsolicited bulk Email (UBE), junk mail, or unsolicited commercial email (UCE), is the practice of sending unwanted e-mail messages, frequently with commercial content, in large quantities to an indiscriminate set of recipients<sup>[1][2]</sup>.

#### 2.2.1 Types of Spam

Some of the most common and general types of spam, encountered during literature review/research are listed:

Unsolicited bulk e-mail (UBE)'s<sup>[2]</sup> basic purpose is to access the personal information of mail user and also attach or append some virus or worm in e-mail. As soon as the user clicks on the unread mail message, this appended script initiates its working automatically. Another sophisticated type of this UBE is in which the mail user is offered outstanding amount of money or prize. For the reception of money/prize a novice user follows the instruction given in the mail and provides his personal details like Credit Card no. etc. Later on, these Credit Card no and other details are used for hacking purposes.

There are different types of techniques used to create UBE.

Blank spam is spam lacking a payload advertisement. Often the message body is missing altogether, as well as the subject line. Still, it fits the definition of spam because of its nature as bulk and unsolicited email. Blank spam may be originated in different ways, either intentional or unintentionally. Blank spam may also occur when a spammer forgets or otherwise fails to add the payload when he or she sets up the spam run. Often blank spam headers appear truncated, suggesting that computer glitches may have contributed to this problem—from poorly-written spam software to shoddy relay servers, or any problems that may truncate header lines from the message body. Some spam may appear to be blank when in fact it is not. An example of this is the VBS. Davinia.B email worm which propagates through messages that have no subject line and appears blank, when in fact it uses HTML code to download other files

Backscatter is a side-effect of e-mail spam, viruses and worms, where email servers receiving spam and other mail send bounce messages to an innocent party. This occurs because the original message's envelope sender is forged to contain the e-mail address of the victim. A very large proportion of such e-mail is sent with a forged header, matching the envelope sender. Since these messages were not solicited by the recipients, are substantially similar to each other, and are delivered in bulk quantities, they qualify as unsolicited bulk email or spam. As such, systems that generate e-mail backscatter can end up being listed on various DNSBLs and be in violation of internet service providers' Terms of Service.

Many spam e-mails contain URLs to a website or websites. These links are used for advertisements of products/sites. There are different types of techniques<sup>[3]</sup> used to create UCE:-

In Image spam, spammers paste an image in the e-mail and link it to the some website, which is undesirable and useless for users. Sometimes it leads to totally out of context website which is again inconvenient and irritating to user.

In Invalid Hyperlinks, spammers provide hyperlinks which are contrary to the text in the mail<sup>[4]</sup>. These links lead to the sites which do not even match with given text descriptions.

Health and Medicine includes advertisements for weight loss, skin care, posture improvement, cures for baldness, dietary supplements, non-traditional medication etc. which can all be bought on-line. Example of such spam is given in figure 2.1.

Subject: Lose up to 19% weight. A new weight loss is here.

Hello, I have a special offer for you...

WANT TO LOSE WEIGHT?

The most powerful weight loss is now available without prescription. All natural Adipren720

100% Money Back Guarantee!

- Lose up to 19% Total Body Weight.
- Up to 300% more Weight Loss while dieting.
- Loss of 20-35% abdominal Fat.



- Reduction of 40-70% overall Fat under skin.

- Increase metabolic rate by 76.9% without Exercise.

- Burns calorized fat.

- Suppresses appetite for sugar.

- Boost your Confidence level and Self Esteem.

Get the facts about all-natural Adipren720: {LINK}

Figure 2.1 Healths and Medicine Spam

IT <sup>[4]</sup> includes offers for low-priced hardware and software as well as services for web site owners such as hosting, domain registration, web site optimization and so forth. Example of IT spam is given in figure 2.2.

Subject: Huge savings on OEM Software. All brand names available now stewardess

Looking for not expensive high-quality software?

We might have just what you need.

Windows	XP	Professional	2002	.....	\$50
Adobe	Photoshop	7.0	.....		\$60
Microsoft	Office	XP	Professional	2002....	\$60
Corel Draw Graphics Suite 11 .....					\$60

And lots more...

Figure 2.2 IT Spam

Personal Finance Spam falls into this category offers insurance, debt reduction services, loans with low interest rates etc. Example of such spam is given in figure 2.3

Subject: Lenders Compete--You Win

Reduce your mortgage payments

Interest Rates are Going Up!

Give Your Family The Financial Freedom They Deserve

Refinance Today & SAVE

\*Quick & EASY

\*CONFIDENTIAL

\*100's Of Lenders

\*100% FREE

\*Get The Lowest Rate

Apply Today! {LINK}

All credit will be accepted

To clear your name from our database please {LINK} or use one of the

options below.

Thank You

Call 1-800-279-7310

Or please mail us at:  
1700 E. Elliot Rd. STE3-C4  
Tempe, AZ. 85283

Figure 2.3 Personal Finance Spam

Education Spam includes offers for seminars, training, and on-line degrees. One such example is in figure 2.4.

Example:

Subject: get a degree from home, Masters, Bachelors or PHD

Call {Phone Num.} to inquire about our degree programs.

Whether you are seeking a Bachelors, Masters, Ph.D. or MBA

We can provide you with the fully verifiable credentials to get your career  
**BACK ON TRACK!**

No testing or coursework required Call: {Phone Num.}

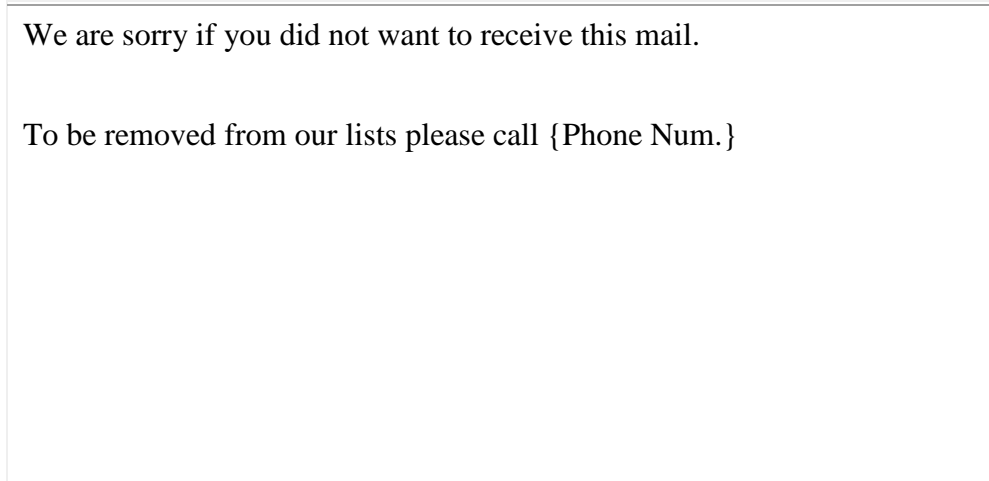


Figure 2.4 A Type of Education Spam

### 2.2.2 Analysis of Different Types of Spam

After studying the above type of spam one can easily classify spam in four major categories: The e-mails having web-links in the message, i.e. through text and through images<sup>[4]</sup> within the message body, the e-mails without body, the e-mails with meaningless text in their body, the e-mails that have some attachments (which actually turn out to be virus or worm) SpamWall would focus on the above types and will try to stop them with minimized false positives and false negatives.

### 2.3 Socket Programming

SpamWall requires capturing an e-mail, applying filtering algorithms and forwarding it to appropriate mailbox folder. To accomplish this task, a sound knowledge of socket programming is required. For this purpose, a thorough study to get basic network programming and higher level network programming was carried out. Some of the key topics related to basic network programming are: Reading from and Writing to sockets, SSL Socket, Creating a datagram and sending it over the network, Properties, Hash Table.

For the advanced network programming, SMTP, POP3, IMAP sockets, Transport, Password Authentication, Message, Session, Store, Folder, and Address are studied.

As the project is developed in java, so few APIs studied, are listed: JDBC, Javamail, Java Email Server (JEC), and EWSJ.

### **2.3.1 JDBC**

JDBC stands for "Java Data Base Connectivity". It is an API<sup>[5]</sup> (Application Programming Interface) which consists of a set of Java classes, interfaces and exceptions and a specification to which both JDBC driver vendors and JDBC developers (like you) adhere when developing applications. The JDBC API is available in the java.sql and javax.sql packages. Following are important JDBC classes, interfaces and exceptions in the java.sql package that will be used in SpamWall: DriverManager, Connection, Statement, ResultSet, and SQL Exception.

For the filtering algorithms (White-listing/Blacklisting and Bayesian Algorithm) a database is required to store the necessary data. For this purpose data base connectivity is required. To attain the given purpose, API (JDBC) to connect to database and store or retrieve the required information is used.

### **2.3.2 Javamail**

The JavaMail API is an optional package (standard extension) for reading, composing, and sending electronic messages. The JavaMail API<sup>[5]</sup> is designed to provide protocol-independent access for sending and receiving messages. This API provides functionalities such as:

Send mail using the JavaMail API, Read mail with the JavaMail API, Deal with sending and receiving attachments, Work with HTML messages, and Search for messages with search terms.

Basically this API is used for fetching mail from the mailboxes. But it should act like a mail server to get the mail contents. But still there are a lot of classes which can help us out to develop SpamWall.

### **2.3.3 JEC (Java Exchange Connector)**

This API<sup>[6]</sup> is used for communicating with Microsoft Exchange Server. It not only provides the simple reception from and forwarding of e-mail to Microsoft Exchange Server but also provides few features: Create folder support, supporting folder types: email, calendar, and task, Attachments support for all handlers, Set Signature support, Set auto-reply support, Move Item support, get all items by URL, Get attachments info without the content support for all handlers.

### **2.3.4 EWSJ (Exchange Web Services Java)**

This API is also used for same purpose as JEC. But there are additional functionalities that this API supports which are: EWSJ<sup>[5]</sup> mailbox sharing (user1 can access user2 mailbox using user1 credentials), EWSJ Impersonation support, PSB (Power Shell Bridge), Ability to Prefer Basic or Digest authentication, Create folder support, Access resources support , Move Item support.

These are not the only limits of these two APIs. These are very powerful APIs for communicating over Microsoft Exchange Server. In SpamWall, these APIs are used for forwarding mail to Exchange Server.

## **2.4 Mail Protocols**

For the development of SpamWall, it is needed to have a sufficient knowledge of e-mail protocols. For this purpose, listed mailing protocols are studied:

### **2.4.1 IMAP Protocol**

IMAP (Internet Message Access Protocol) is a standard protocol for accessing e-mail from your local server. IMAP is a client/server protocol in which e-mail is received and held for you by your Internet server. As this requires only a small data transfer this works well even over a slow connection such as a modem. Only if you request to read a specific email message will it be downloaded from the server. You can also create and manipulate folders or mailboxes on the server, delete messages etc. IMAP's ability to access messages (both new and saved) from more than one computer has become extremely important as reliance on electronic messaging and use of multiple computers increase, but this functionality cannot be taken for granted: the widely used Post Office Protocol (POP) works best when one has only a single computer, since it was designed to support "offline" message access, wherein messages are downloaded and then deleted from the mail server. This mode of access is not compatible with access from multiple computers since it tends to sprinkle messages across all of the computers used for mail access.

Key features for IMAP include:

- Allow message access and management from more than one computer.
- Allow access without reliance on less efficient file access protocols.
- Provide support for "online", "offline", and "disconnected" access modes
- Support for concurrent access to shared mailboxes
- Client software needs no knowledge about the server's file store format.

### **2.4.2 POP3 Protocol**

The POP (Post Office Protocol 3) protocol provides a simple, standardized way for users to access mailboxes and download messages to their computers. When using the POP protocol

all your email messages will be downloaded from the mail server to your local computer. You can choose to leave copies of your emails on the server as well. The advantage is that once your messages are downloaded you can cut the internet connection and read your email at your leisure without incurring further communication costs. On the other hand you might have transferred a lot of message (including spam or viruses) in which you are not at all interested at this point.

POP supports simple download-and-delete requirements for access to remote mailboxes (termed maildrop in the POP RFC's). Although most POP clients have an option to leave mail on server after download, e-mail clients using POP generally connect, retrieve all messages, store them on the user's PC as new messages, delete them from the server, and then disconnect. Other protocols, notably IMAP, (Internet Message Access Protocol) provide more complete and complex remote access to typical mailbox operations. Many e-mail clients support POP as well as IMAP to retrieve messages; however, fewer Internet Service Providers (ISPs) support IMAP.

A POP3 server listens on well-known port 110. Encrypted communication for POP3 is either requested after protocol initiation, using the STLS command, if supported, or by POP3S, which connects to the server using Transport Layer Security (TLS) or Secure Sockets Layer (SSL) on well-known TCP port 995.

Available messages to the client are fixed when a POP session opens the maildrop, and are identified by message-number local to that session or, optionally, by a unique identifier assigned to the message by the POP server. This unique identifier is permanent and unique to the maildrop and allows a client to access the same message in different POP sessions. Mail is retrieved and marked for deletion by message-number. When the client exits the session, the mail marked for deletion is removed from the maildrop.



### 2.4.3 SMTP Protocol

The SMTP (Simple Mail Transfer Protocol) protocol is used by the Mail Transfer Agent (MTA) to deliver your email to the recipient's mail server. The SMTP protocol can only be used to send emails, not to receive them. Depending on your network / ISP settings, you may only be able to use the SMTP protocol under certain conditions.

While electronic mail servers and other mail transfer agents use SMTP to send and receive mail messages, user-level client mail applications typically only use SMTP for sending messages to a mail server for relaying. For receiving messages, client applications usually use either the Post Office Protocol (POP) or the Internet Message Access Protocol (IMAP) or a proprietary system to access their mail box accounts on a mail server.

SMTP is a text-based protocol, in which a mail sender communicates with a mail receiver by issuing command strings and supplying necessary data over a reliable ordered data stream channel, typically a Transmission Control Protocol (TCP) connection. An SMTP session consists of commands originated by the SMTP client and corresponding responses from the SMTP server by which the session is opened, session parameters are exchanged, the recipients are specified and possibly verified, and the message is transmitted, before the session is closed. The originating host is either an end-user's email client, functionally identified as a mail user agent (MUA), or a relay server's mail transfer agent (MTA).

SMTP was designed as an electronic mail transport and delivery protocol, and as such it is used between SMTP systems that are operational at all times. However, it has capabilities for use as a mail submission protocol for email clients (split user-agent) that do not have the capability to operate as MTA. Such agents are also called message submission agents (MSA), sometimes also

referred to as mail submission agents. They are typically end-user applications and send all messages through a smart relay server, often called the outgoing mail server, which is specified in the programs' configuration. A mail transfer agent, incorporated either in the e-mail client directly or in the relay server, typically determines the destination SMTP server by querying the Domain Name System for the mail exchanger (MX record) of each recipient's domain name. Conformant MTAs fall back to a simple address lookup (A record) of the domain name when no mail exchanger is available. In some cases an SMTP client, even a server, may also be configured to use a smart host for delivery. The SMTP client typically initiates a Transmission Control Protocol (TCP) connection to the SMTP server on the well-known port designated for SMTP, port number 25.

SMTP is a delivery protocol only. It cannot pull messages from a remote server on demand. Other protocols, such as the Post Office Protocol (POP) and the Internet Message Access Protocol (IMAP) are specifically designed for retrieving messages and managing mail boxes. However, the SMTP protocol has a feature to initiate mail queue processing on a remote server so that the requesting system may receive any messages destined for it (cf. Remote Message Queue Starting). POP and IMAP are preferred protocols when a user's personal computer is only intermittently powered up, or Internet connectivity is only transient and hosts cannot receive message during off-line periods.

#### **2.4.4 Relay Protocols**

In computer networking, the Message Session Relay Protocol (MSRP) is a protocol for transmitting a series of related instant messages in the context of a communications session. An application initiates the session with the Session Initiation Protocol (SIP).

For example, relay protocol can be used within a SIP session: to do Instant Messaging in a one-to-one or one-to-many mode, to transfer file attachment, to do some Image Sharing based on prior exchange of capabilities between the user endpoints.

## **2.5 Configuration of Exchange Server**

As SpamWall will be deployed on MS Exchange Server 2003, for testing and training purposes, a working exchange server is required. College's Exchange server is not available so an Exchange Server was configured. To gain this objective, few steps required to configure Exchange Server are: Install MS Windows Server 2003, Install IIS (Install Information Server) with SMTP component, Install DNS Services i.e. Set DNS Settings if DHCP is used and Otherwise, only enter DNS Servers, Create a new Active Directory i.e. After selecting a new domain controller, create a new domain tree and forest of domain trees, In full DNS name - enter the name of your domain that you have registered, Accept the NetBIOS equivalent of your domain, Select the locations you want to store the databases, Accept the location of the system folder , Set the permissions and create a password to administer the directory, Install MS Exchange Server 2003, Add mail users.

## **2.6 Spam Filtering Algorithms**

There are different types of filtering e-mails <sup>[7]</sup>. Different types of algorithms deal with different types of spam. Some major of these techniques are given:

### **2.6.1 Spam Firewall**

Spam firewalls <sup>[7]</sup> are installed by companies to identify and prevent spam from entering their networks. These are highly effective when implemented on networks of 100 to 20,000 employees as they help to minimize workload from the server.

## 2.6.2 Using Blacklists

Blacklists or blocklists<sup>[8]</sup> are lists of IP addresses, domain names, email addresses or content of the headers or the body, or some combination of these different types that can be used to help identify spam. A special subset of IP address and domain name lists exist which can be queried using DNS, which are called DNS Blackhole Lists or *DNSBLs*. Blacklists can be unverified and cause “collateral damage”; their criteria for listing may not be clear.

This approach is also known as RBL (Real-time Blackhole List). It's maintained by system administrators who, using various spam detection tools, report bad-behaving IP addresses (e.g. open relays or hosts that were detected to spend undesired email, have no registered DNS record, etc.). This information goes into a central database, and is then shared by those who want to use it. So rather than trying to filter each email separately, here all email coming from a blacklisted IP is rejected as soon as the connection is established.

There are many RBLs available. Some are more aggressive (blocking whole net blocks), whereas others are more flexible. One way to deal with false-positives here is to try to query several RBLs and then make a decision based on whether they all agree or not.

Organizations such as SpamCop and MAPS maintain a blacklist database which bans a certain entity from sending emails. The blacklisting can be by IP address, person, company or domain.

## 2.6.3 Using White-Lists

White-lists<sup>[8]</sup> are exact opposites of blacklists. White-lists can be used by communities where members must be pre-approved in order to send and receive emails. It's based on the nightclub concept "If your name isn't on the list, you aren't getting in".

Most spam filtering systems use blacklists, where mail from a certain list of email addresses or matching a certain list of text patterns is rejected or otherwise filtered. These lists take a lot of time and effort to maintain and in the end still don't work very well. The way Whitelist-based filtering works is: you create a list of addresses of people you expect to receive mail from, and filter anything that is not from them into a separate low-priority mailbox that you check once a week or once a month or something. The white listing for mail filters is based on the logic that if a person sends an email the person in general will accept mail from that recipient. By collecting these TO addresses from outgoing mail exclusion database can be generated.

#### **2.6.4 Spam Database**

This method involves feedback from the user community to report spam when they receive it. When enough users report a message as spam, it trips a red flag which marks the message as spam. Spam blockers use this information to block the offending message from further users.

#### **2.6.5 Bayesian Filtering**

Bayesian filtering <sup>[9]</sup> is a recent development employed by spam blocker software and it appears to be very effective for detecting and removing spam at the personal level. Bayesian spam filtering is a statistical technique of e-mail filtering. It makes use of a naive Bayes classifier to identify spam e-mail. Bayesian filtering is an intelligent approach to classifying mail because it examines all aspects of a message, as opposed to keyword checking that classifies a mail as spam on the basis of a single word. For example: not every email that contains the word 'free' and 'cash' is spam. The Bayesian method would find the words 'cash' and 'free' interesting but

it would also recognize the name of the business contact who sent the message and thus may classify the message as legitimate; it allows words to 'balance' each other out.

Particular words have particular probabilities of occurring in spam email and in legitimate email. The filter doesn't know these probabilities in advance, and must first be trained so it can build them up. To train the filter, the user must manually indicate whether a new email is spam or not. For all words in each training email, the filter will adjust the probabilities that each word will appear in spam or legitimate email in its database. For instance, Bayesian spam filters will typically have learned a very high spam probability for the words "Viagra" and "refinance", but a very low spam probability for words seen only in legitimate email, such as the names of friends and family members.

After training, the word probabilities (also known as likelihood functions) are used to compute the probability that an email with a particular set of words in it belongs to either category. Each word in the email contributes to the email's spam probability, or only the most interesting words. This contribution is called the posterior probability and is computed using Baye's theorem. Then, the email's spam probability is computed over all words in the email, and if the total exceeds a certain threshold (say 95%), the filter will mark the email as a spam. Like in any other spam filtering technique, email marked as spam can then be automatically moved to a "Junk" email folder, or even deleted outright. Some software implements quarantine mechanisms that define a time frame during which the user is allowed to review the software's decision. The initial training can usually be refined when wrong judgments from the software are identified (false positives or false negatives). That allows the software to dynamically adapt to the ever evolving nature of spam. The formula used by the software to determine is derived from Bayes' theorem. <sup>[8][9]</sup>

$$P(A|B) = \frac{P(B|A) P(A)}{P(B)}.$$

In the spam filtering field above formula can be expressed as:

$$\Pr(\text{spam}|\text{words}) = \frac{\Pr(\text{words}|\text{spam}) \Pr(\text{spam})}{\Pr(\text{words})}$$

In other words, the probability of being spam for a content in the case that it has a set of certain words is equal to the multiplication of the conditional probability of each word to be spam in the probability of begin spam, divided by the probability of the existence of these words in any content.

### 2.6.6 Heuristic Filtering

Heuristic simply means speculative. It works by examining messages for certain high risk patterns which might be flagged as spam<sup>[9]</sup>. Modern heuristic engines allow user overrides and have to update its filtering rules regularly as spam evolves. Heuristic filtering works by subjecting email messages through thousands of pre-defined rules against the message envelope, header and content. Each rule assigns a numerical score to the probability of the message being spam. The result of the final equation is known as the Spam Score.

The spam score is then measured against the user's desired level of spam sensitivity - whether low, medium or high sensitivity. Setting a higher level of sensitivity leads to more spam being captured, but has the adverse effect of filtering legitimate emails as spam. This is known as a false-positive. Heuristic engines however suffer from a main drawback. As spammers learn and adopt according to the filtering rules employed, so too must the heuristic database. This has kept a cat and mouse game of playing catch up and is therefore important that the spam filter software

supports updating its heuristic database as part of its feature. Additionally, heuristic engines tend to be very error prone and good spam filter software should not only have a low false-positive rate, but should also allow custom rules to be defined.

### **2.6.7 Challenge Response Spam Filtering**

When using a challenge response system, each email address must be authorized to deliver email to you. You can either authorize these email addresses manually or have the challenge response email server take care of that for you.

A Challenge-response (or C/R) system is a type of spam filter that automatically sends a reply with a challenge to the (alleged) sender of an incoming e-mail. In this reply, the sender is asked to perform some action to assure delivery of the original message, which would otherwise not be delivered. The action to be performed is typically one that can be performed once relatively effortlessly, but needs great effort if performed in large numbers, in this way effectively filtering out spammers. Challenge-response systems only need to send challenges to unknown senders. Senders that have previously performed the challenging action, or who have previously been sent e-mail(s) to, would be automatically white listed. C/R systems attempt to provide challenges that can be fulfilled easily for legitimate senders and non-easily for spammers. Two characteristics that differ between legitimate senders and spammers are exploited in order to achieve this goal. Legitimate senders have a valid return address, while spammers usually forge a return address. This means that most spammers won't get the challenge, making them automatically fail any required action. Spammers send e-mail in large quantities and have to perform challenging actions in large numbers, while legitimate senders have to perform it at most once for every new e-mail contact.



## **2.7 Analysis of Filtering Techniques**

SpamWall has been developed on the basis of Spam Firewall. The techniques which are implemented in this system are: Blacklists Algorithm, White-lists Algorithm, and Bayesian Algorithm (which also incorporates the Heuristics Algorithm) <sup>[10]</sup>.

In addition to the above techniques, another technique is required to implement named as link-checker. This technique will check the links given in the message (whether they are in text or images). Challenge Response Filtering is not incorporated in SpamWall because of its waiting time requirement.

## **2.8 Structure of Firewall**

The firewall (sometimes referred to as a bastion host) is a subsystem of computer software and hardware that intercepts data packets before allowing them into or out of a Local Area Network (LAN). A firewall makes decisions on whether or not data is allowed to pass based upon a security policy. For each packet of data, the firewall compares known components of the Packet to a security rule set and decides if the packet should be allowed to pass. In addition, a firewall may have security rules that involve altering the packet in some basic ways before passing the data. With a sensible security policy and a security rule set designed to implement that policy, a firewall can protect a LAN from attacks.

### **2.8.1 Firewall Types**

A true firewall is the hardware and software that intercepts the data between the Internet and your computer. All data traffic must pass through it, and the firewall allows only authorized data to pass into the corporate network. Firewalls are typically implemented using one of four

primary architectures: Packet Filters, Circuit-level Gateways, Application Proxies, and Network Address Translation.

Each type uses different packet filtering techniques from each other, but still considered part of the same architecture. SpamWall is also based on the architecture of firewall.

## **2.9 Java Mail Server (JES)**

Java Email Server (JES) is a well known, open source, complete and production ready email server written in Java by Eric Daugherty. Java Mail Server is a Java SMTP and POP3 e-mail server. This project exists for everyone who is interested in running their own email server quickly and easily. JES provides an easy to use and reliable email server that can be quickly and easily setup. JES is much appropriate as it doesn't waste long time waiting to receive test mails, no connection problems, mail account is full of test mails, no installation is needed, easy to configure and provide accounts management (you can add as many accounts as you want). Salient features of JES are: Starting and stopping mail server from Eclipse, You can test your mails service written in any language (Java, PHP etc) under Eclipse, It launches a local mail server: almost no time between sending and receiving mails, No server installation and easy configuration.

## **2.10 Summary**

This chapter discussed the literature overview. It described the proposed approaches along with the mechanisms used and the problems that are found in these approaches due to which they cannot be considered very efficient for the spam attacks detection. It included a detailed description about all types of spam attacks found and the algorithms and techniques available to filter mail.

## Chapter 3

# Requirements Specifications

### 3.1 Introduction

Requirements analysis encompasses those tasks that go into determining the needs or conditions to meet for a new or altered product, taking account of the possibly conflicting requirements of the various stakeholders, such as beneficiaries or users. Systematic requirements analysis is also known as requirements engineering. Requirements analysis of the SpamWall is done in this chapter.

#### 3.1.1 Purpose

The purpose of this chapter is to specify the requirements of the degree project ‘SpamWall’. SpamWall is an application to analyze e-mail and filter it as spam or ham. This software will integrate several spam filtering algorithms to filter spam from e-mail. This will also be linked with an existing antivirus to scan attachments in the mail. The purpose of SpamWall is to facilitate the MCS mail server with spam free environment.

#### 3.1.2 Conventions

When writing this document it was inherited that Functional Requirements of the SpamWall Project will be at high level and given more importance than those of Non-Functional Requirements and all functional requirements have same priority. First there is presented an overall view about SpamWall and then all features and functions are analyzed in detail.

This requirement document contains general information about SpamWall and its key features. It describes in detail all that SpamWall needs to work properly. The rest of the chapter is divided into sub-chapters for better understanding: In “**Overall Description**” an overall

description of SpamWall is provided. First product perspective is presented with product features and main functions. Then follow user classes and characteristics, operating environments that SpamWall supports as well as design and implementation constraints. After all that user documentation is presented and will provide you with more details about each feature's technology, In " **System Features**" most important features are presented with detailed description and requirements, In "**External Interface Requirements**" and communication interfaces are described, In "**Other Nonfunctional Requirements**" requirements about safety and performance are presented.

## **3.2 Overall Description**

Primary purpose of SpamWall is to filter mail and forward to exchange server. It is also linked with antivirus to scan attachments for viruses. Overall Description is further divided into sub-topics for elaboration.

### **3.2.1 Product Perspective**

SpamWall is self-contained product (application software) which has a primary task to filter unwanted emails called spam. This software will be linked to an existing antivirus to scan the attachments within an email. SpamWall will be installed on a separate machine, which will receive all the incoming emails and after filtering and scanning, will forward the mail to MS Exchange Server.

### **3.2.2 Product Features**

SpamWall provides the user with the functionalities such as: Enable SpamWall, Disable SpamWall, Filter mail, Scan Attachments, Forward mail, System Log. Through an interface administrator can enable the SpamWall and it will start giving its services. Through an interface

administrator can disable the SpamWall and it will stop giving its services. When SpamWall is enabled, it will analyze all incoming mails and filter them as ham or spam. SpamWall will be linked with any existing antivirus to scan all the attachments within the mail. After filtering and scanning, SpamWall will forward the mail to MS Exchange Server as ham or Spam. Through an interface SpamWall will provide the log of the system i.e. the time when SpamWall is set enable, number of spam caught etc.

### 3.2.3 Top Level Data Flow Diagram

Top level Data Flow Diagram is given in figure 3.1. It describes the flow of data in SpamWall. Email is captured by Receiver Module and after applying filtering algorithms, it is forward to Exchange Server.

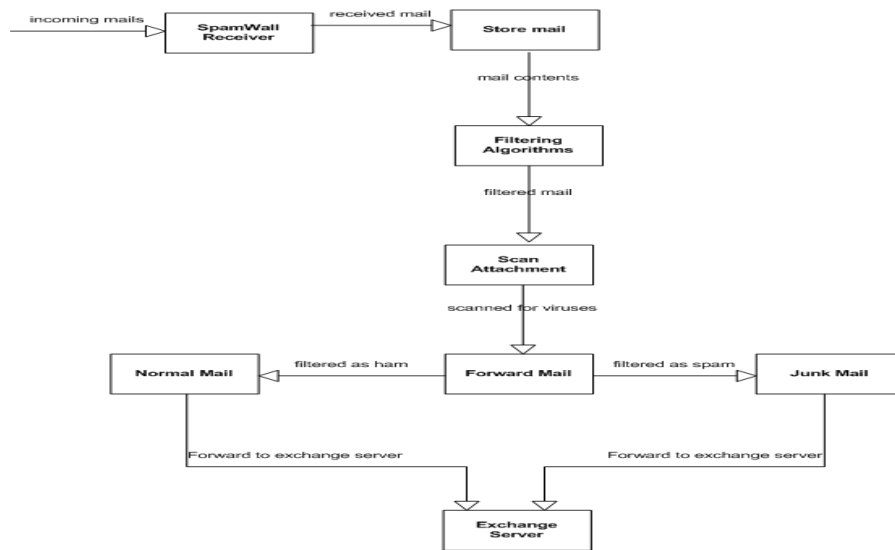


Figure 3.1 Top Level DFD

Receiving Module receives mail from internet. This module will use SMTP, POP3, MIME, IMAP protocols for receiving mail. Then the incoming mail is decomposing into two components i.e. To/From (sender/Receiver) and Message Body. Then these components are

forwards to Filtering Module. Filtering Module filters mail and forwards to Exchange Server using Forward Module.

### **3.2.4 User Classes and Characteristics**

All users who use the domain ‘**mcs.edu.com**’ will be using this application, which include students, faculty members, system administrator and other staff members. There is no extra or technical knowledge is required for this application. An ordinary user who can use mail applications like yahoo mail or Gmail etc can use this application. Administrator can enable and disable the application through simple interface. Users will have very little interaction with the application. In case of any false positive or false negative, Administrator can blacklist or white list the sender.

### **3.2.5 Operating Environment**

SpamWall should run on Windows Operating Systems preferably Windows XP. JVM will also be required to be installed in the system. A P4 machine with 3GHz microprocessor and 512MB ram is enough to operate the SpamWall. An existing antivirus will be required to scan the attachments in the mail and to ensure the security of the system itself. Nothing more than these is required for a fully functional SpamWall.

### **3.2.6 Design and Implementation Constraints**

There are few design and implementation constraints of SpamWall. In developing SpamWall timing requirement has been considered to enhance the efficiency. Algorithms for filtering are applied in an order to minimize the processing time and enhance efficiency and performance. It is assumed that processing time will be less than 100miliseconds. JVM (Java Virtual Machine) is essential to be installed in the system in order to operate the SpamWall as all

the functionality is implemented in JAVA. An up to date antivirus is also required for the scanning of attachments in the mail and for the security of system.

### **3.2.7 User Documentation**

A user manual will be provided with the SpamWall Application to guide the users. It will have all the necessary information and method of installation of application to help the users.

## **3.3 System Features**

System features are organized by use cases and functional hierarchy so that the main functions of the system will be understandable.

### **3.3.1 Enable SpamWall**

An administrator must enable the SpamWall through a simple interface to get its services. Administrator is just required to click on '**Enable**' button given in the interface of SpamWall and application will start functioning: System must be connected to internet, JVM must be installed. It is assumed that there will not be any error condition in the software, however in case of any hardware failure application will stop functioning.

### **3.3.2 Disable SpamWall**

An administrator can disable the SpamWall through a simple interface to stop its services. Administrator is just required to click on '**Disable**' button given in the interface of SpamWall and application will stop functioning: System must be connected to internet, JVM must be installed, and SpamWall is enabled. It is assumed that there will not be any error condition in the software, however in case of any hardware failure application will stop functioning.

### **3.3.3 Filter Mail**

When SpamWall is enabled, it will analyze all incoming mails and filter them as ham or spam. SpamWall will use following algorithms to analyze the mail: Blacklists algorithm, Whitelist algorithm, Link-checker algorithm, Heuristics algorithm, Bayesian algorithm. When SpamWall is enabled, application will automatically apply filtering algorithms to all incoming mails without any user interference: System must be connected to internet, JVM must be installed, and SpamWall must be enabled. It is assumed that there will be minimum numbers of false positives and false negatives but due to the limitations of algorithms and increasingly new types of spam application may misclassify mail. In this case user can blacklist or white-list the sender.

### **3.3.4 Scan Attachments**

SpamWall will be linked with any existing antivirus to scan all the attachments within the mail, when it is enabled. When SpamWall is enabled, application will automatically scan for the attachments within the mail by the help of some existing antivirus: System must be connected to internet, JVM must be installed, SpamWall must be enabled, and SpamWall must be linked with an existing antivirus. Scanning of attachments in the mail and correctness of analyzing attachments depends upon the strength of antivirus.

### **3.3.5 System Log**

Through an interface SpamWall will provide the log of the system. i.e. The time when SpamWall is set enable, number of spam caught etc. It will also give information about the nature of spam i.e. blacklist spam, white-list spam etc. Administrator is just required to click on '**Log**' button given in the interface of SpamWall and application will give the log information: System must be connected to internet, JVM must be installed, and SpamWall must be enabled. It



is assumed that there will not be any error condition in the software, however in case of any hardware failure application will stop functioning.

### **3.3.6 Forward Mail**

After filtering and scanning, SpamWall will forward the mail to MS Exchange Server as ham or Spam. After filtering and scanning, SpamWall will automatically forward the mail to MS Exchange Server as ham or Spam without any interference of user: System must be connected to internet, JVM must be installed, and SpamWall must be enabled. It is assumed that there will not be any error condition in the software, however in case of any hardware failure application will stop functioning.

## **3.4 Other Nonfunctional Requirements**

Apart from functional requirements, non-functional requirements are also very important in development of any project. Few non-functional requirements of SpamWall are listed:

### **3.4.1 Performance Requirements**

In developing SpamWall timing requirement has been considered to enhance the efficiency. Algorithms for filtering are applied in an order to minimize the processing time and enhance efficiency and performance. It is assumed that processing time will be less than 100miliseconds.

### **3.4.2 Security Requirements**

In order to make the system secure, an up to date antivirus must be installed in the system. This antivirus is dual purpose as it is also required to scan the attachments in the mail.

### **3.4.3 Software Quality Attributes**

Applications of filtering algorithms are prioritized in a way that light algorithms are applied first, minimizing the processing time and increasing performance and efficiency. Application will be available and functional after it is enabled. It is reusable and portable to any Windows machine. It is assumed application will minimize the number of false positives and false negatives however correctness of scanning and analyzing attachments depends upon strength of antivirus.

### **3.5 Summary**

In this chapter, the requirement analysis of the system is discussed. It also includes the functional requirements of the system and the basic functionality the system must provide. SpamWall must capture mail from all senders and must classify it as spam or ham and forward to Exchange Server.

## **Chapter 4**

# **Software Design**

### **4.1 Introduction**

System design is a very important phase in the software development process. The succeeding implementation phase is performed taking into consideration the design constraints. This chapter begins by presenting the high level design of the project showing the main modules of the system without including much detail. Next the low level design is incorporated elucidating the modules identified in the high level design. It is then followed by the data flow diagram of the project. Class diagram is also included focusing on the implemented classes, their attributes and their relationships with each other. The design process is facilitated after the choice of a suitable process model.

### **4.2 Design Consideration**

In the development of SpamWall, there were a few constraints which needed to be considered and are stated as:

#### **4.2.1 General Constraints**

SpamWall will operate under these constraints:

##### **4.2.1.1 Timing Requirements in SpamWall**

In developing SpamWall timing requirement has been considered to enhance the efficiency. Algorithms for filtering are applied in an order to minimize the processing time and enhance efficiency and performance. It is assumed that processing time will be less than 100milliseconds.

#### **4.2.1.2 JVM Installation**

JVM (Java Virtual Machine) is essential to be installed in the system in order to operate the SpamWall as all the functionality is implemented in JAVA.

#### **4.2.1.3 Antivirus**

SAVSE (Semantic Anti-Virus Scan Engine) is also required for the scanning of attachments in the mail and for the security of system. It should be installed on exchange server and would work in connection with the SpamWall.

### **4.3 Goals**

The basic goals during implementation are accuracy is preferred over speed, the KISS principle ("Keep it simple stupid!"), to emphasis on speed, minimum usage of memory, and follow common conventions, so that user does not feel inconvenient in new environment. Using these implementation goals this application is efficient in speed and accuracy as well.

### **4.4 Development Methods**

For SpamWall, Agile software methodology 'Scrum' is used. The main roles in Scrum are the "Scrum Master", who maintains the processes. The "Product Owner", who represents the stakeholders. The "Team", a group of people who do the actual analysis, design, implementation, testing, etc.

During each "sprint", typically a two to four week period (with the length being decided by the team), the team creates a product increment. The set of features that go into a sprint come from the product "backlog," which is a prioritized set of high level requirements of work to be done. Which backlog items go into the sprint is determined during the sprint planning meeting.

During this meeting, the Product Owner informs the team of the items in the product backlog that he or she wants completed. The team then determines how much of this they can commit to complete during the next sprint. After a sprint is completed, the team demonstrates the use of the software. A key principle of Scrum is its recognition that during a project the customers can change their minds about what they want and need, and that unpredicted challenges cannot be easily addressed in a traditional predictive or planned manner. In this way scrum welcomes the changes in requirements. In our project Scrum Master is the supervisor.

#### **4.5 Architectural Strategies**

This application follows the architecture of a firewall or simply it acts like a firewall as denoted by figure 1.1. It receives the email from internet cloud, applies filtering algorithms on it, classifies it as spam or ham and forwards it to exchange server. Application has been developed using Java mail server, which is open source and in this particular application acts like a firewall. This server has been configured and modified according to our requirements. Mail coming from outside (internet cloud) is first passed through SpamWall, classified as good or bad mail by the application and then forwarded to Exchange server. The main objective of using this strategy is to minimize the load on mail Exchange server and direct putting spam in Junk folder and normal mail in Inbox.

#### **4.6 Main Components (Modules) of SpamWall:**

SpamWall comprises of three modules, i.e. Receiving Module, Filtering Module and Forwarding Module as represented in the figure 4.1.

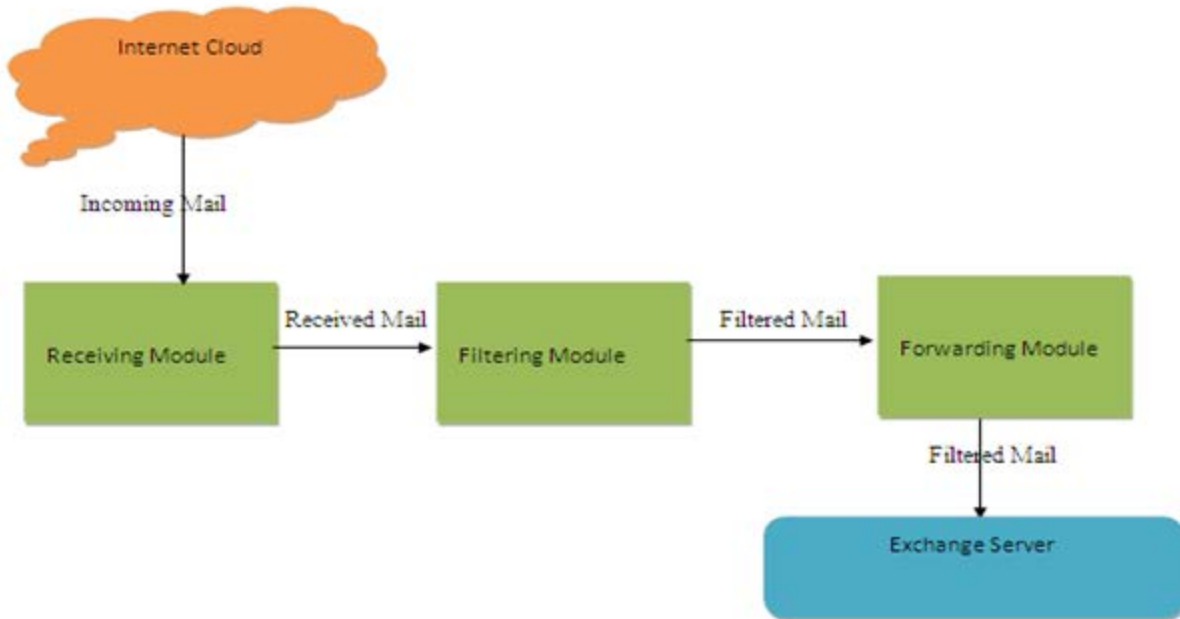


Figure 4.1 High-Level Diagram

From the internet cloud a mail is received by Receiving Module. This module accepts email and stores it in a variable and tosses it to filtering module. Filtering Module gets the email, applies filtering algorithms and flags it and tosses to Forwarding Module. Forwarding Module forwards the mail to Exchange Server. The detailed working and responsibilities of the modules is explained as:

#### 4.6.1 Receiving Module:

Receiving Module receives mail from internet. This module will use SMTP, POP3, MIME, IMAP protocols for receiving mail. Then the incoming mail is decomposed into two components i.e. To/From (sender/Receiver) and Message Body. Then these components are forwarded to Filtering Module.

#### **4.6.2 Filtering Module:**

Filtering Module is then decomposed into four components which are Blacklist Algorithm/White-List Algorithm, Scan Attachment, Link-Checking Algorithm and Bayesian Algorithm. To/From part is first sent to White-List Algorithm, and if the sender is in contacts or White-List database, no more algorithm is applied on mail and mail is forwards to Forwarding Module with flag set. Else Blacklist, Attachment scan, Link-Checking, Bayesian Algorithms are applied in sequence. If in the course, mail is found guilty, no more algorithm in applied and forwards to Forwarding Module with flag set. These algorithms are applied sequentially. Any email which is marked as spam does not have to face rest of the filtering algorithms. It gives an extra edge for efficiency of application.

#### **4.6.3 Forwarding Module:**

It receives mail with flag set from Filtering Module and forwards the mail to specific folder in exchange server.

In integration mostly composition was used. This was due to flexibility and reusability requirement of the design. In case of change, only specific part will be modified without having to change rest of the integrating modules.

## 4.7 Low-Level Diagram

Detailed design is described in figure 4.2. All individual modules are described in the diagram.

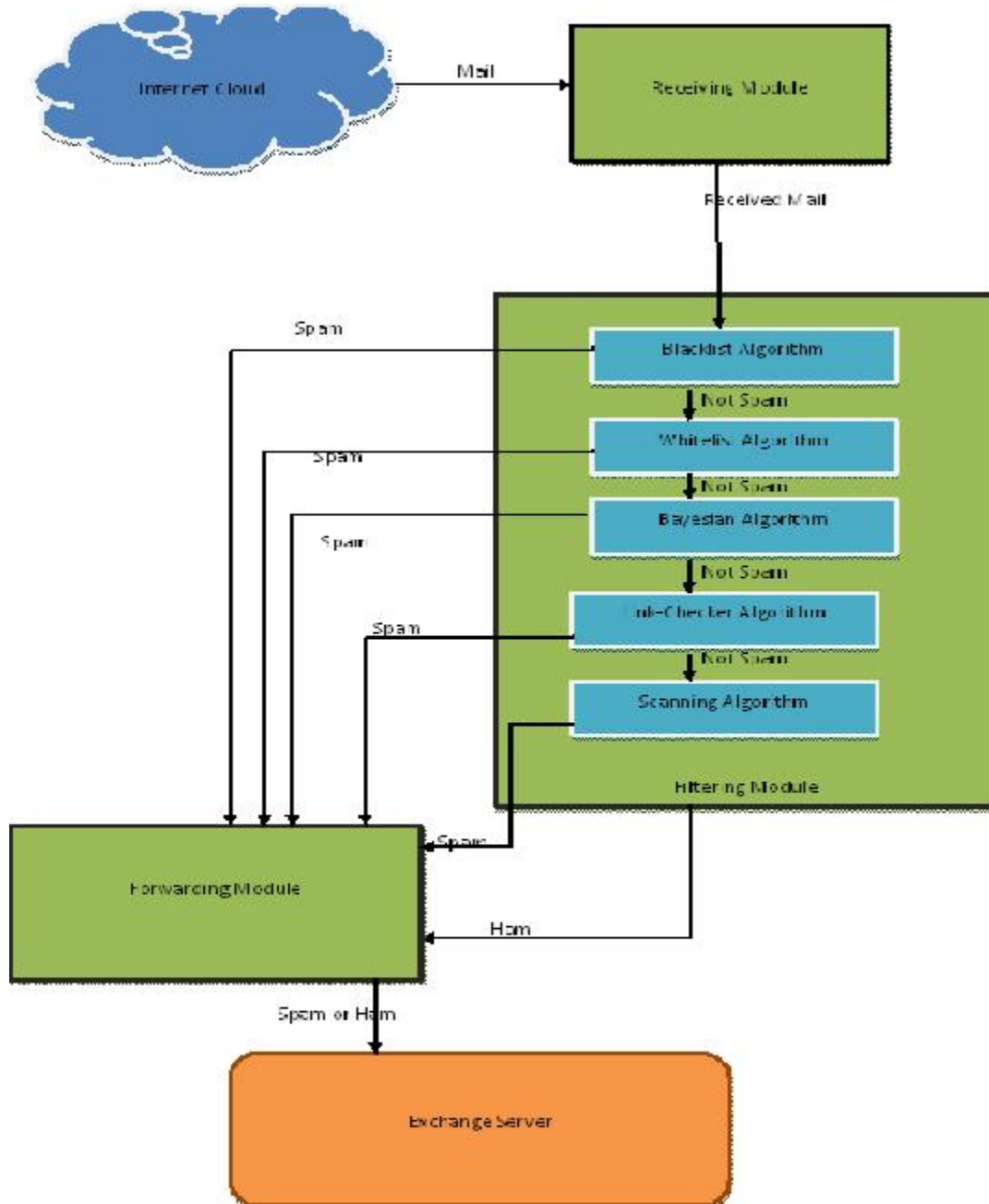


Figure 4.2 Low Level Design



Figure 4.2 describes the low-level design of SpamWall. Receiving Module receives mail from internet. This module will use SMTP protocol for receiving mail. Then the incoming mail is decomposed into few components i.e. To/From (sender/Receiver), Message Body and attachment. Then these components are forwarded to different Filtering Module. Filtering Module is composed of four components which are Blacklist Algorithm/White-List Algorithm, Scan Attachment, Link-Checking Algorithm and Bayesian Algorithm. To/From part is first sent to White-List Algorithm, and if the sender is in contacts or White-List database, no more algorithm is applied on mail and mail is forwards to Forwarding Module as ham. Blacklist module also takes from address and compares it with it database to check sender is blacklisted or not. In former case mail is directly sent to forwarding module as spam and in later case mail is considered as ham by this algorithm and further filtering is done. Bayesian Algorithms is applied on the mail which passes blacklist test and contents are analyzed to classify it spam or ham, if it is segregated as spam, it is forward to forwarding module else link-checker module take over and checks for suspicious hyperlinks. At the end scanning module scans the attachments for spywares. These algorithms are applied sequentially. Any email which is marked as spam does not have to face rest of the filtering algorithms. It gives and extra edge for efficiency of application.

## 4.8 Use Case

The Use Case Diagram of the application is presented in figure 4.2.

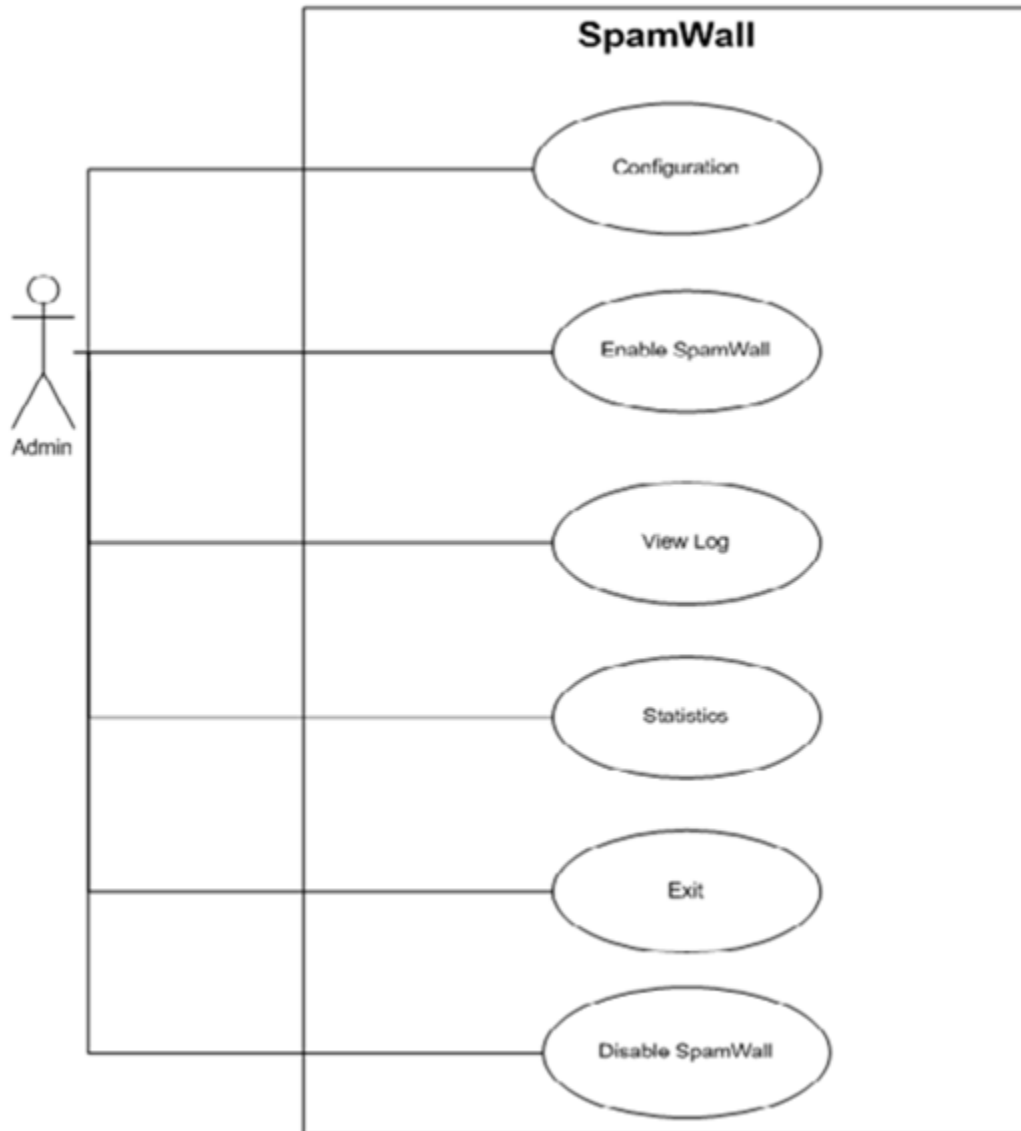


Figure 4.3 Use-Case Diagram

The primary actor is administrator and he is capable of using all the use cases. He can enable services of application through a user interface, disable services of application, exit the

application, configure the SpamWall, View statistics for analysis and view log of the system. All these cases can be handled through a Graphical User Interface.

## 4.9 Class Diagram

Class diagram has been divided into two parts because of having many classes, which can't be explained in single diagram. One part is explained in figure 4.4a.

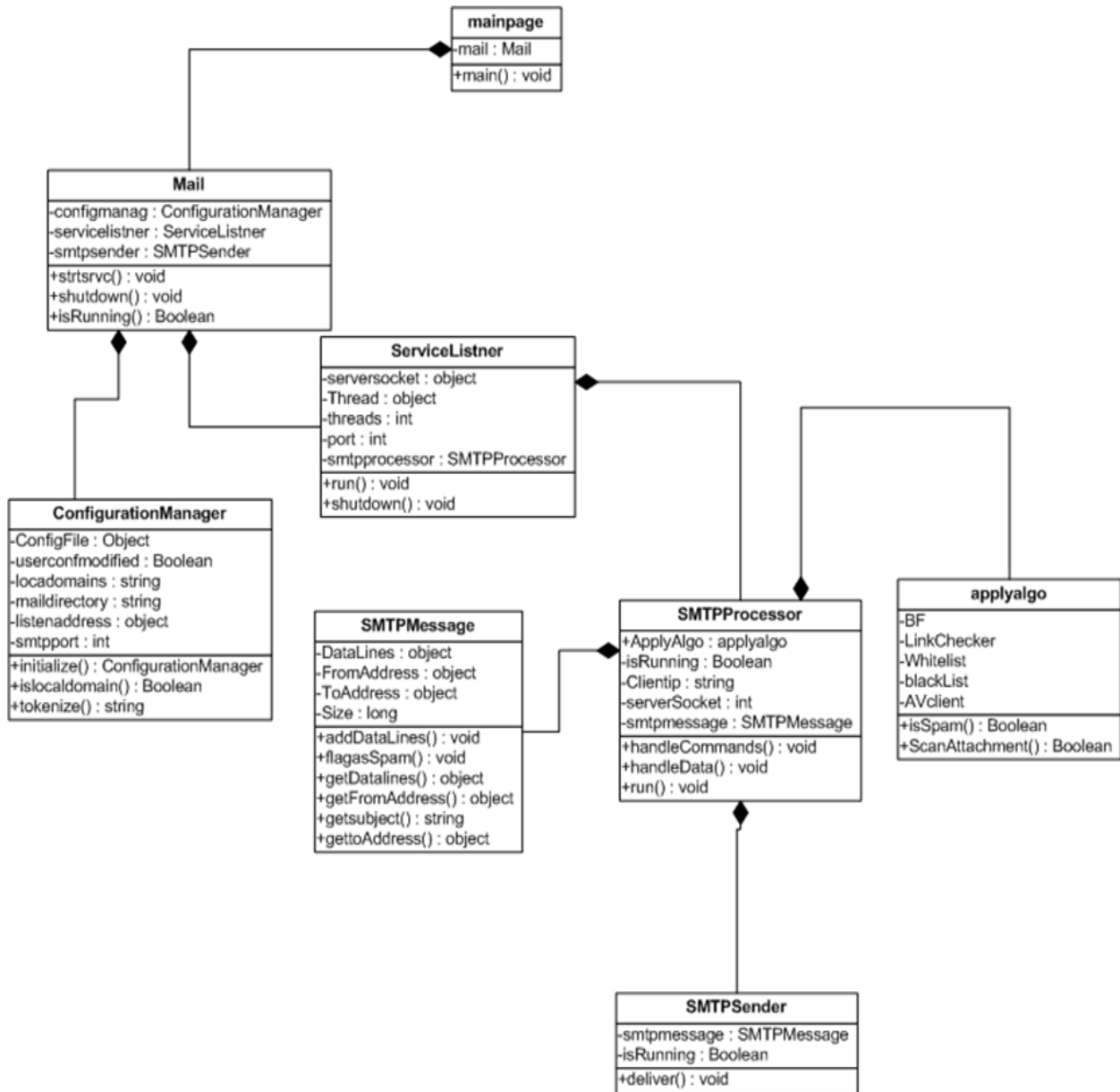


Figure4.4a Class Diagram

Figure 4.4a describes the relationship among different classes. Mainpage is the class which contains the object of Mail class, which further contains the objects of other classes. Composition is mostly used in the classes, which makes it easy to invoke the methods of other classes. In second part of diagram relationship of algorithms applied on mail is described. It is described in figure 4.4b.

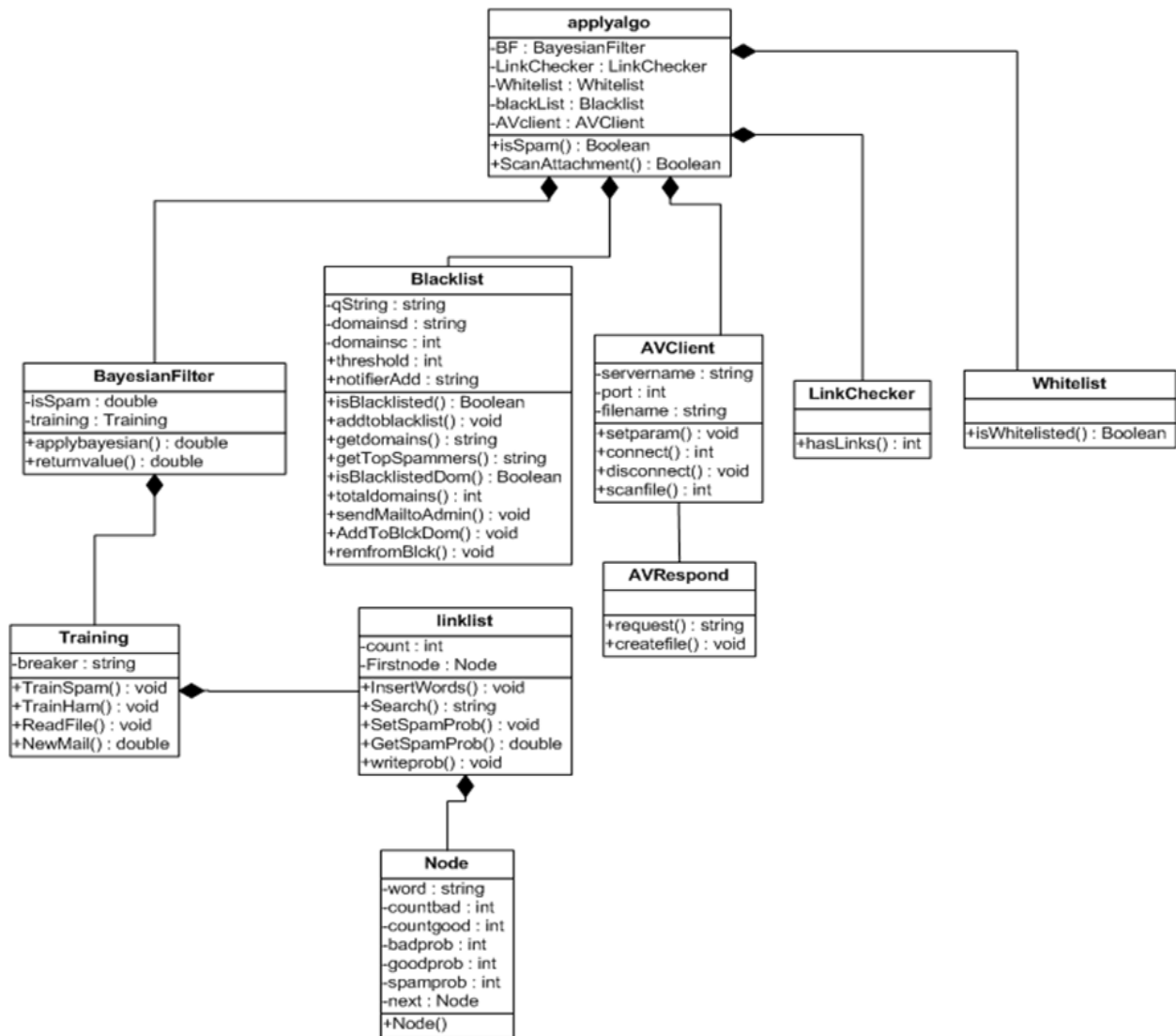


Figure 4.4b Class Diagram

In figure 4.4b relationship of filtering algorithms is described. Applyalgo is the main class which contains the objects of filtering algorithms and invokes their respective functions for filtering the mail. AVClient class is used for scanning of attachments in the mail.

### 4.10 Sequence Diagram

The sequence diagram of the Application is given in the figure 4.5.

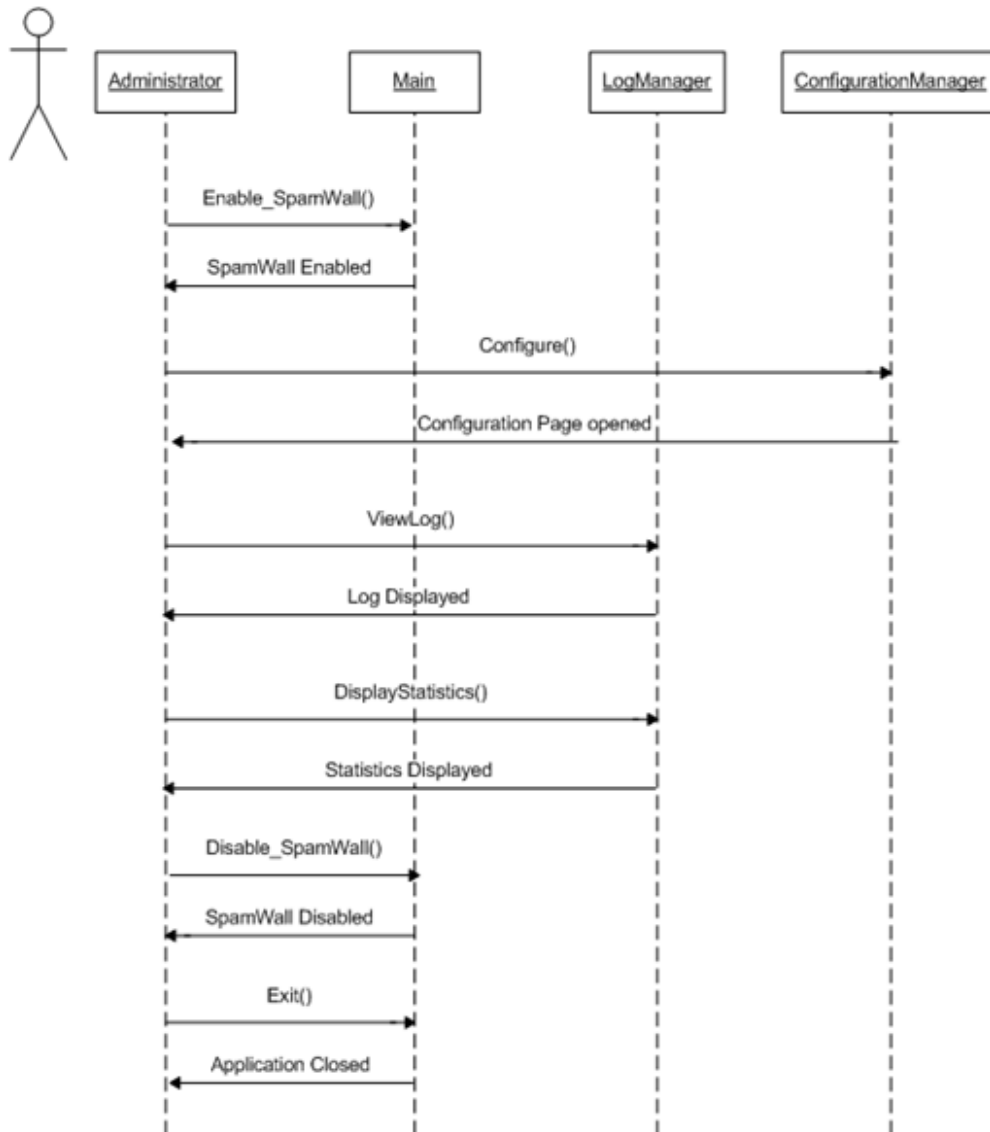


Figure 4.5 Sequence Diagram

For each Use Case a sequence is defined for the flow of data. In case of the ViewLog system gets to application log and gives a response to administrator. For the case of Configuration, application communicates with Configuration Manager to make changes in the configuration of application. Rest of the Use cases is entertained by System's main Class.

#### 4.11 Dataflow Diagram

The Dataflow Diagram of the application is represented in figure 4.6.

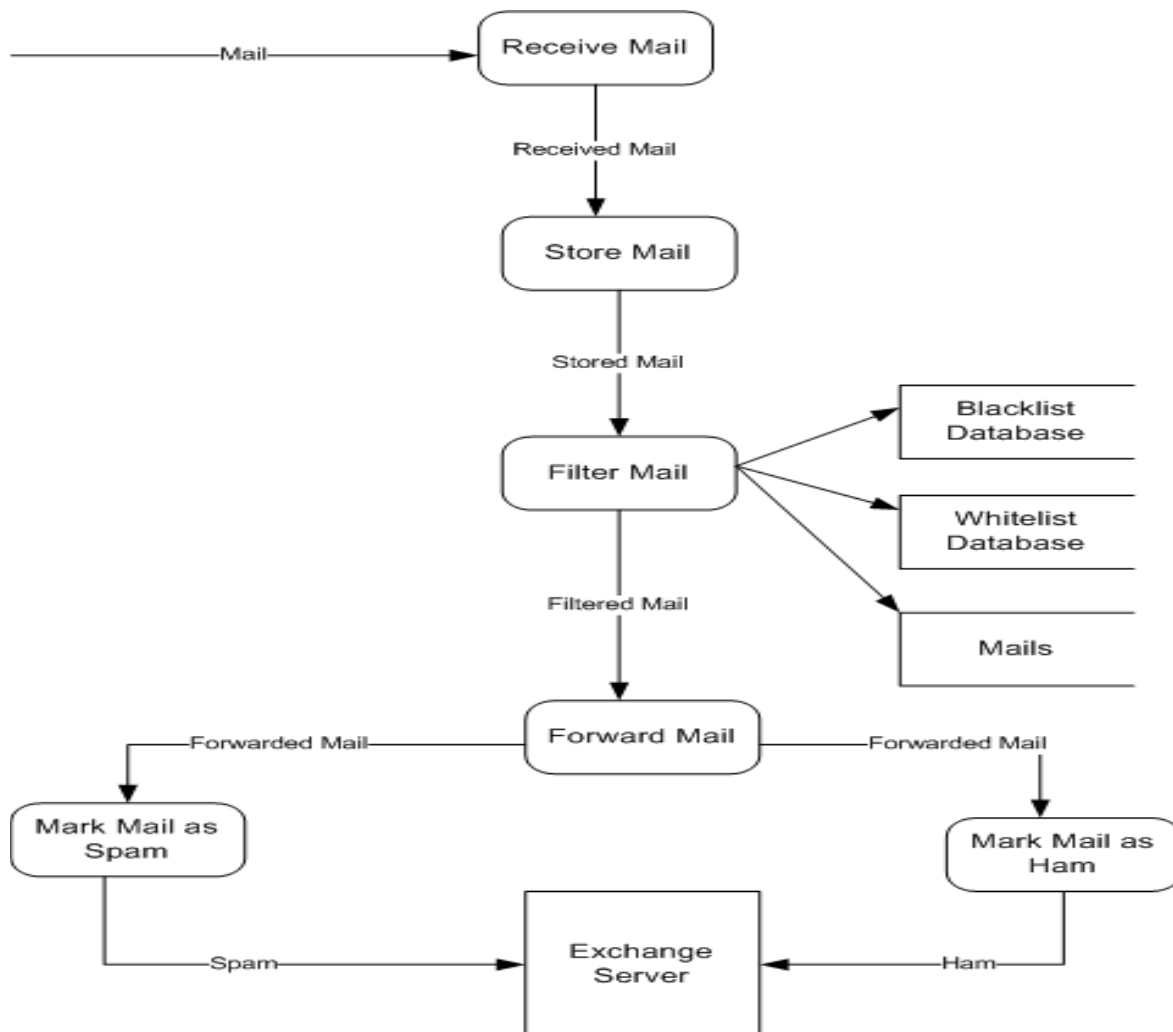


Figure 4.6 Data Flow Diagram

A data-flow diagram (DFD) is a graphical representation of the "flow" of data through an information system. Main steps followed in SpamWall are system will receive mail from internet; receiving Module will store mail and decompose into two parts, filtering algorithms will be run on both parts and checked for spam, attachment, if any, will be scanned through existing antivirus, forwarding module set the flag and marked mail as Spam and Normal Mail and then forwards mail to the specific folder in exchange server.

#### **4.12 Summary**

This chapter presented the architecture for SpamWall using filtering algorithms. It has incorporated the high level design, low level design, data flow diagram and class diagram, and the sequence diagram of the system. The main modules identified are receiver module, filtering module, scanning module and forwarding module. The system design chapter included details that are very important in comprehending well the upcoming implementation chapter.

## **Chapter 5**

# **Implementation**

### **5.1 Introduction**

This chapter presents the implementation details of the project. The coding is done in the object oriented language JAVA in the netbeans environment. As illustrated in the prior chapter, the major modules of the system are JES (Java Email Server), Bayesian filtering Algorithm, Blacklist/Whitelist filtering algorithm, Link checker algorithm, Email Scanner and the GUI.

### **5.2 Implementation Language**

The implementation language which has been chosen for the project is JAVA. It has been preferred over other languages because of several reasons e.g. it is a platform independent language which enhances the system's portability. Also it is an object oriented language which makes it simple to visualize the objects in real life. Project required Socket programming, which is also a significant reason to use JAVA. JES (Java Email Server) has been used in the project which is open source and has been implemented in Java. All the filtering algorithms were developed in Java.

### **5.3 Implementation of JES**

JES <sup>[1]</sup>(Java Email Server) is open source email server in java. It has all the functionalities of an email server. This application to receives mail from outside and forwards it to exchange server. JES (Java Email Server) was used and modifies according to needs of the project. In this particular application JES acts like a firewall as it receives the email before MS Exchange



Server. Filtering Algorithms integrated with JES filter the Email and then JES forwards it to exchange server.

SMTP\_processor.java class is used to receive email from internet cloud. Read () function in this class is used to read the email. Applyalgo.java contains objects of all the filtering algorithm classes. SMTP\_processor.java class calls all the methods of filtering algorithms through Applyalgo.java. Different parts of mail are passed to filtering algorithms for classification. From address is passed to blacklist and white-list algorithms to check if sender is blacklisted or white-listed. Similarly body is passed to Bayesian class and attachment to AVClient.java for scanning. These classes apply filtering algorithms on mail and after classification mail sender.java sends mail as spam or ham to Exchange Server.

#### **5.4 Implementation of Bayesian**

After SMTP\_processor.java class receives the email, Filtering algorithms are applied in series on email to classify it as spam or ham. BayesianFilter.java, Training.java, linklist.java and node.java are classes used for implementation of Bayesian Filtering Algorithm. Node.java is used to store the words of email. Linklist.java applies the algorithm on mail to classify the content.

$$P(A|B) = \frac{P(B|A) P(A)}{P(B)}.$$

In the spam filtering field above formula can be expressed as the following.

$$\Pr(\text{spam}|\text{words}) = \frac{\Pr(\text{words}|\text{spam}) \Pr(\text{spam})}{\Pr(\text{words})}$$

Training.java is used to train the algorithm. It maintains a database of sample mails which are either spam or ham and classifies the new incoming mail by comparing it to the database of sample mails.

### **5.5 Implementation of Blacklist Filtering Algorithm**

bListAlgo.java class is used to classify mail on the bases of blacklisting algorithm. isBlacklist () method checks if sender is in blacklist database and add\_to\_black\_list () adds a sender into blacklist database, if sender is a spammer. It also finds out top spam senders from the database and top spam sending domains, which are used in statistics.

### **5.6 Implementation of White-List Filtering Algorithm**

wListAlgo.java class is used to classify mail on the bases of whitelisting algorithm. isWhitelisted () method is used to check if sender is in whitelist and simply marks the mail as ham. No further filtering is done on such sender.

### **5.7 Implementation of Link Checker Algorithm**

LinkChecker.java class is used to check the suspicious hyperlinks in email. It detects the hyperlinks in email and if a match is found from given database of suspicious hyperlinks, mail is considered as spam.

### **5.8 Implementation of Email Scanner Algorithm**

AVClient.java, AVRespond.java and VirusException.java are the classes used to scan the attachment in the email. AVClient.java connects to SAVSE (Semantic Anti-Virus Scan Engine) and sends email along with attachments for scanning purpose. AVRespond.java class returns either true or false depending upon whether attachment or header of mail contains any virus etc or not. VirusException.java caters error conditions and throws exception if any occurs.

## 5.9 Implementation of GUI

Administrator can interact with the SpamWall through GUI. Main page is shown in figure 5.1a.



Figure 5.1a GUI

Main page provides administrator with the functionalities described in the figure 5.1a. It provides user friendly environment. Administrator just has to use mouse to click the buttons. Enable SpamWall calls the `startsvc()` method from Mail class and application starts its services. Similarly Disable SpamWall calls the method `shutdown()` and SpamWall stops services. View Log shows a file which contains all the Log of SpamWall i.e. Time of incoming message, it is spam or ham etc. Similarly View Statistics shows statistics. It also gives the option of Configure and Minimize.

Statistics page gives the information of statistics as in figure 5.1b.

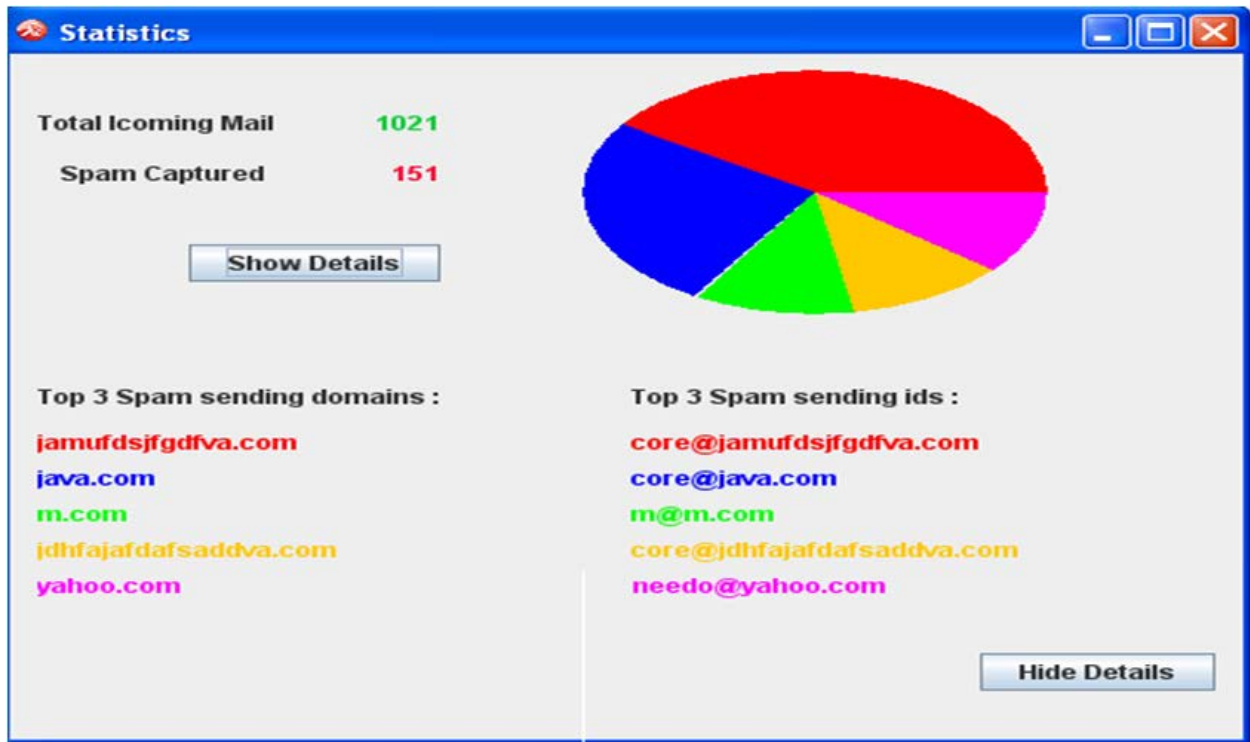


Figure 5.1b GUI

Figure 5.1b shows total number of incoming mails and number of spam in those mails. It shows the E-mail addresses of top Spam Senders and Spam Sending Domains. This information is also shown by the help of a pie chart. SpamWall automatically update all the information in statistics page.

Administrator can configure the settings using Configuration page as in figure 5.1c.

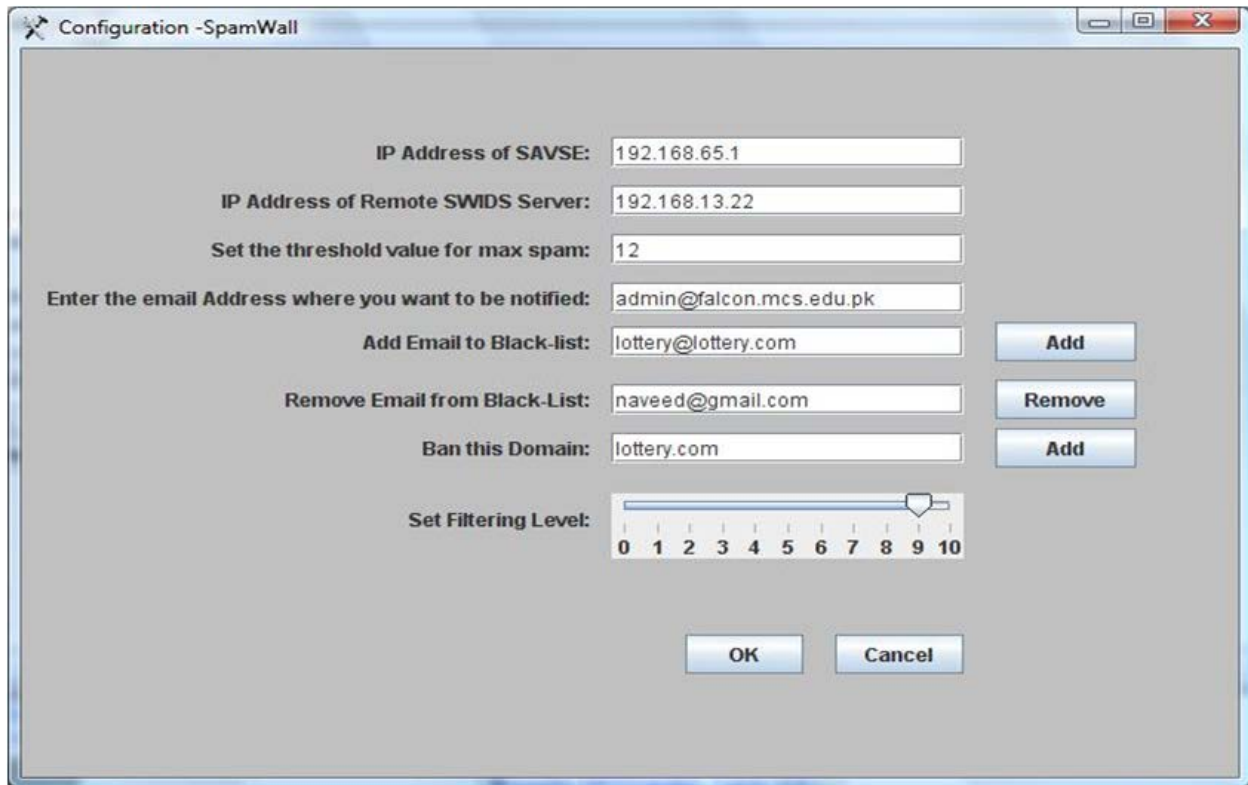


Figure 5.1c GUI

As the application scans the attachments using SAVSE (Semantic Anti-Virus Scan Engine) which is installed on MS Server. Administrator can set the IP Address of MS Server, where SAVSE is installed. SpamWall can be integrated with SWIDS Server to inform the receiver about incoming mail. So its IP address is also required and can be set by administrator.

Threshold value is number of Spam a spammer can maximum send and after that administrator can take appropriate action against him. Similarly administrator can blacklist any sender or

domain and also remove the blacklisted sender from blacklist. Administrator can also control the filtering level to control false positives and false negatives.

To enhance user friendly nature a tray icon has also been added as in figure 5.2.

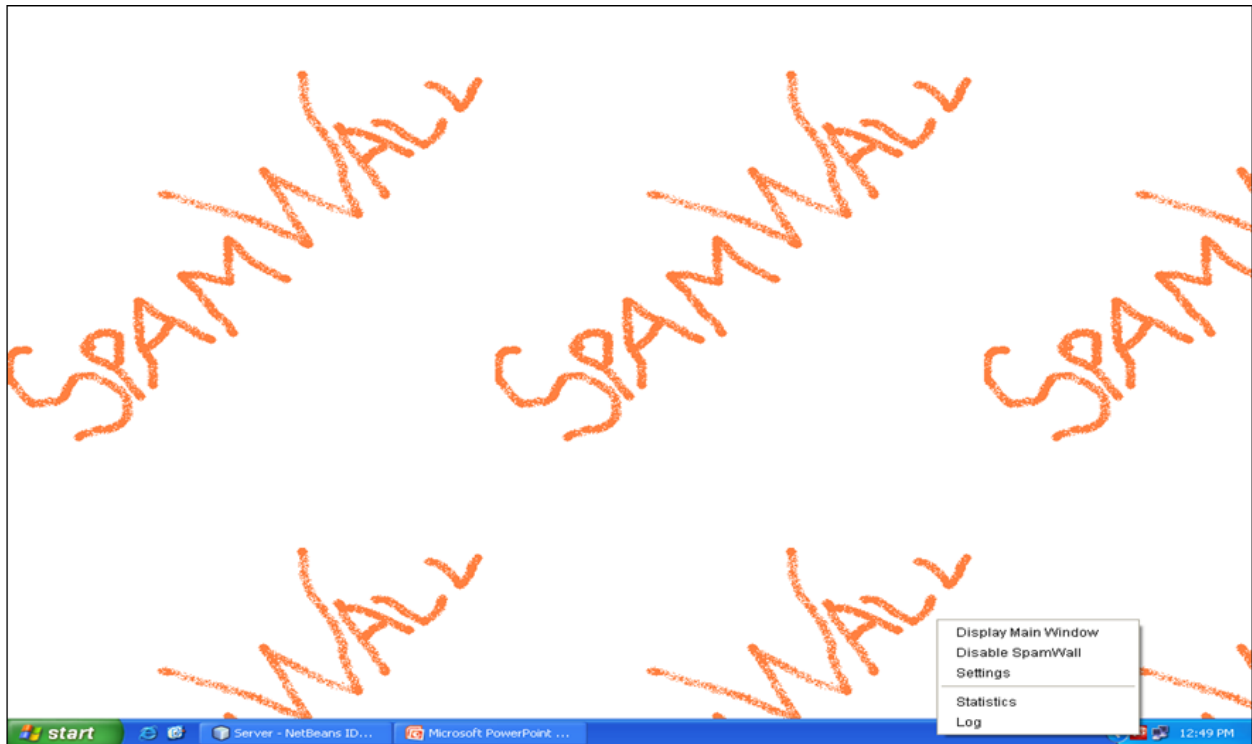


Figure 5.2 Tray Icon

Tray icon shown in figure 5.2 is used to minimize the main window and administrator can access all the features of GUI through a small icon on task bar. It does not provide any extra functionality but just facilitates administrator.

## 5.10 Summary

This chapter incorporated the details of the classes implemented. The classes have been distributed among the three basic components of the system which are JES, Filtering Algorithms

i.e. Bayesian Filtering Algorithm, blacklisting Filtering Algorithm, Link Checker Algorithm and the Mail Scanner module. JAVA has been used as the programming language for the project due to its object-oriented and platform independent nature.

## **Chapter 6**

# **Testing & Analysis**

### **6.1 Introduction**

Software testing and analysis is one of the most crucial phases of software development life-cycle. This can be termed as an element of a broader topic that is referred to as 'Verification and Validation' (V&V). Verification refers to the set of activities that ensure that software correctly implements a specific function. Validation refers to the different set of activities that ensures that the software that has been built is traceable to customer requirements.

### **6.2 Validation and Verification**

Validation and verification is intended to be a systematic and technical evaluation of the system and its processes. To effectively deal with the increased complexity and functionality, systems need practical techniques that can help improve software quality using the validation and verification process.

SpamWall has been tested for validation by giving it different set of content of email and getting the desired outputs. In verification testing it was assured that software meets all functional, behavioral, and performance requirements.

### **6.3 Unit Testing**

In computer programming, unit testing is a procedure to validate that individual units of source code are working properly. A unit is the smallest testable part of an application. Unit testing concentrates on each unit of the software as implemented in source code. The goal of unit testing is to isolate each part of the program and show that the individual parts are correct. In this application each module as developed is individually tested. The each function performed on the module is tested. Each operation and function is individually tested so as to check for possible errors that could occur.

### **6.4 Integration Testing**

Integration testing is the phase of software testing in which individual software modules are combined and tested as a group. It follows unit testing and precedes system testing. In integration testing focus is on design and the construction of software architecture. All the modules have been combined and tested to ensure that they work according to the user requirements.

### **6.5 Black Box Testing**

Black box testing takes an external perspective of the test object. These tests can be functional or non-functional, through usually functional. The test designer selects the valid and invalid input and determines the correct output. There is no knowledge of the test object's internal structure. The requirements established as part of software requirements analysis, are validated against the software that has been constructed.

SpamWall has been fully validated as per meeting user requirement. It provides final assurance that the software meets all functional, behavioural, and performance requirements.



## **6.6 System Testing**

In software testing phase overall system is tested as a whole. System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. The purpose of integration testing is to detect any inconsistencies between the software units that are integrated together. System testing is a more limiting type of testing; it seeks to detect defects within the system as a whole.

In SpamWall user has no interaction with application, only administrator interacts and no inputs required by him so problem of giving wrong inputs is no longer in this application. Administrator is provided with Graphical User interface to select from given options i.e. Enable or Disable SpamWall so no chance of giving uncertain inputs.

## **6.7 Summary**

The system is tested thoroughly. All the modules have been tested for validation and results were found 95% accurate in all the algorithms except for Bayesian filtering, where few false positives and false negative were found and were tried to minimize to the maximum possible extent. All the modules have been integrated and tested thoroughly and in most of the cases results were found according to the requirements. On entering the required inputs system behaves according to the user requirement and presents the desired results to the user.

## **Chapter 7**

# **Conclusion & Future Work**

### **7.1 Introduction**

This chapter describes the possible future enhancements in the project. As it is a research project, there are a lot of enhancements that can be done in order to make the system more and more effective and efficient.

### **7.2 Enhancement in Filtering Algorithms**

The Filtering Algorithms that are being developed cater only about few prominent types of spamming attacks. In future, the algorithms can be increased by catering many other spam attacks. Field of spamming keeps evolving and by the passage of time new spamming techniques are discovered so new techniques will be required to block such attacks. Application did not incorporate Challenge Response Filtering because of its waiting time requirement. It can be incorporated by optimizing the implantation and minimizing waiting time.

### **7.3 Enhancement in the Compatibility**

A number of enhancements can be made in the compatibility issue. At present SpamWall is compatible with MS Exchange Server only. It can be modified to be used with other email servers.

#### **7.4 Enhancement in the Functionality**

Functionality of SpamWall can be greatly enhanced and many other features can be incorporated apart from filtering email. One of the features is to send an instant message to the receiver informing about incoming email. When email is received in inbox of a user, another application “A messaging System (like SMS)” integrated with SpamWall can send message to the user cell phone informing about the received email.

#### **7.5 Increase in Actions**

The system at present provides the Enable, Disable and Whitelist/Blacklist action. In future, other actions can be added to the system. These actions may include the DELETE action giving the administrator the privilege to delete the malicious code from the different portions of an attachment thus neutralizing it and letting it pass the application without causing any harm to the information at the user end.

Another action can be that of DISCARD. This action privileges the administrator to discard email if found malicious thus stopping it to reach the user end.

#### **7.6 Interaction of End User**

One of the enhancements can be to provide user an interaction with the application. At present user has no interaction with the application. SpamWall can be modified to provide user with several options like: Blacklist or Whitelist the mail sender, control the level of filtering, move to junk or inbox etc.

#### **7.7 Conclusion**

SpamWall is an anti-spam software application/tool to filter spam from original email using best email filtering algorithms. Algorithms used in SpamWall are: Blacklist Algorithm,

White list Algorithm, Bayesian filtering Algorithm, Hyperlinked based Algorithm. Attachments contained within email can also be a security threat as it can have malwares i.e. Spyware, Viruses etc. To cater this problem SpamWall is linked with existing anti-virus software to scan for suspicious attachments. Given an email SpamWall will check for: Text messages (content classification), Empty messages, suspicious web links, linking with existing antivirus to scan the attachments for malwares i.e. Spyware and Viruses etc.

It is a server side system which will scan all the traffic coming towards MS Exchange server for the presence of any Spam, suspicious hyperlink or any suspicious attachment.

## **7.8 Summary**

This chapter described the future enhancements in the system and how is it going to affect the performance and efficiency of the system.

**APPENDIX A**

**USER MANUAL**

## Main Form

The main form which is displayed is shown in the Figure 1.



Figure 2

To enable SpamWall, administrator needs to click on the “Enable SpamWall” button. This button makes the application available and ready to accept data from external mail server.

To stop the services of the system, administrator needs to click on the “Disable SpamWall” button. This will make the application stop its services and shut down all the services of the system.

To view the log of the SpamWall, administrator needs to click on “View Log” button. It opens the notepad editor of the windows and displays the log file of the SpamWall as depicted in the Figure 2.

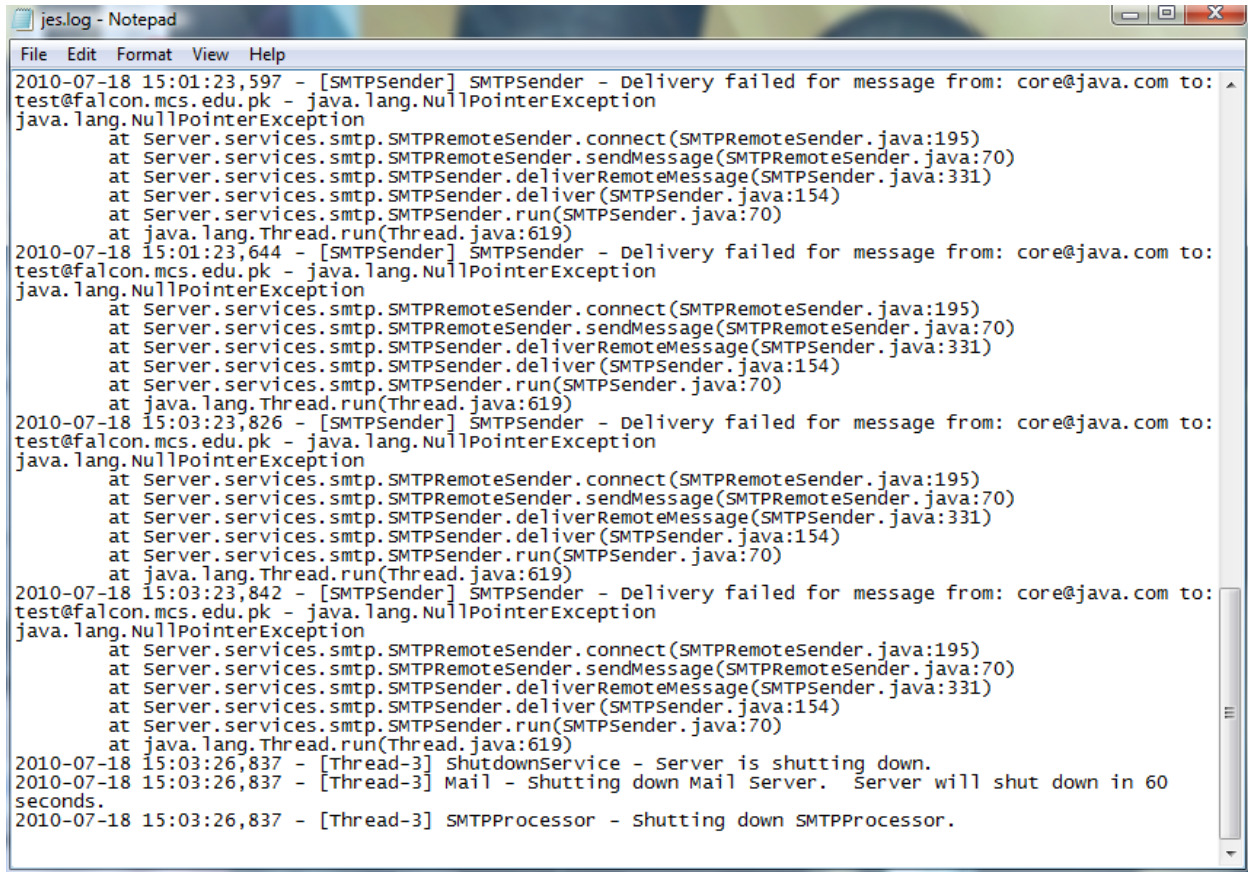
To configure the SpamWall, administrator needs to click on the “Statistics” button. It will result in the opening of another form with the title “Statistics- SpamWall” as depicted in Figure 3.

To configure the SpamWall, administrator needs to click on the “Configure” button. It will result in the opening of another form with the title “Configuration- SpamWall” as depicted in Figure 4.

To hide this main form, administrator needs to click on “Minimize” button. Administrator can view this form again by clicking on “View Main Window” in tray icon as depicted in Figure 5.

## **Log View**

The system log is presented in Figure 2.



```
jes.log - Notepad
File Edit Format View Help
2010-07-18 15:01:23,597 - [SMTPSender] SMTPSender - Delivery failed for message from: core@java.com to:
test@falcon.mcs.edu.pk - java.lang.NullPointerException
java.lang.NullPointerException
    at Server.services.smtp.SMTPRemoteSender.connect(SMTPRemoteSender.java:195)
    at Server.services.smtp.SMTPRemoteSender.sendMessage(SMTPRemoteSender.java:70)
    at Server.services.smtp.SMTPSender.deliverRemoteMessage(SMTPSender.java:331)
    at Server.services.smtp.SMTPSender.deliver(SMTPSender.java:154)
    at Server.services.smtp.SMTPSender.run(SMTPSender.java:70)
    at java.lang.Thread.run(Thread.java:619)
2010-07-18 15:01:23,644 - [SMTPSender] SMTPSender - Delivery failed for message from: core@java.com to:
test@falcon.mcs.edu.pk - java.lang.NullPointerException
java.lang.NullPointerException
    at Server.services.smtp.SMTPRemoteSender.connect(SMTPRemoteSender.java:195)
    at Server.services.smtp.SMTPRemoteSender.sendMessage(SMTPRemoteSender.java:70)
    at Server.services.smtp.SMTPSender.deliverRemoteMessage(SMTPSender.java:331)
    at Server.services.smtp.SMTPSender.deliver(SMTPSender.java:154)
    at Server.services.smtp.SMTPSender.run(SMTPSender.java:70)
    at java.lang.Thread.run(Thread.java:619)
2010-07-18 15:03:23,826 - [SMTPSender] SMTPSender - Delivery failed for message from: core@java.com to:
test@falcon.mcs.edu.pk - java.lang.NullPointerException
java.lang.NullPointerException
    at Server.services.smtp.SMTPRemoteSender.connect(SMTPRemoteSender.java:195)
    at Server.services.smtp.SMTPRemoteSender.sendMessage(SMTPRemoteSender.java:70)
    at Server.services.smtp.SMTPSender.deliverRemoteMessage(SMTPSender.java:331)
    at Server.services.smtp.SMTPSender.deliver(SMTPSender.java:154)
    at Server.services.smtp.SMTPSender.run(SMTPSender.java:70)
    at java.lang.Thread.run(Thread.java:619)
2010-07-18 15:03:23,842 - [SMTPSender] SMTPSender - Delivery failed for message from: core@java.com to:
test@falcon.mcs.edu.pk - java.lang.NullPointerException
java.lang.NullPointerException
    at Server.services.smtp.SMTPRemoteSender.connect(SMTPRemoteSender.java:195)
    at Server.services.smtp.SMTPRemoteSender.sendMessage(SMTPRemoteSender.java:70)
    at Server.services.smtp.SMTPSender.deliverRemoteMessage(SMTPSender.java:331)
    at Server.services.smtp.SMTPSender.deliver(SMTPSender.java:154)
    at Server.services.smtp.SMTPSender.run(SMTPSender.java:70)
    at java.lang.Thread.run(Thread.java:619)
2010-07-18 15:03:26,837 - [Thread-3] ShutdownService - Server is shutting down.
2010-07-18 15:03:26,837 - [Thread-3] Mail - Shutting down Mail Server. Server will shut down in 60
seconds.
2010-07-18 15:03:26,837 - [Thread-3] SMTPProcessor - Shutting down SMTPProcessor.
```

Figure 3

This window uses system notepad to show the log of SpamWall. Here all the information about start and end time of services, including forwarding of email is stored.



## Statistics Form

Statistics of the SpamWall are shown in Figure 3.

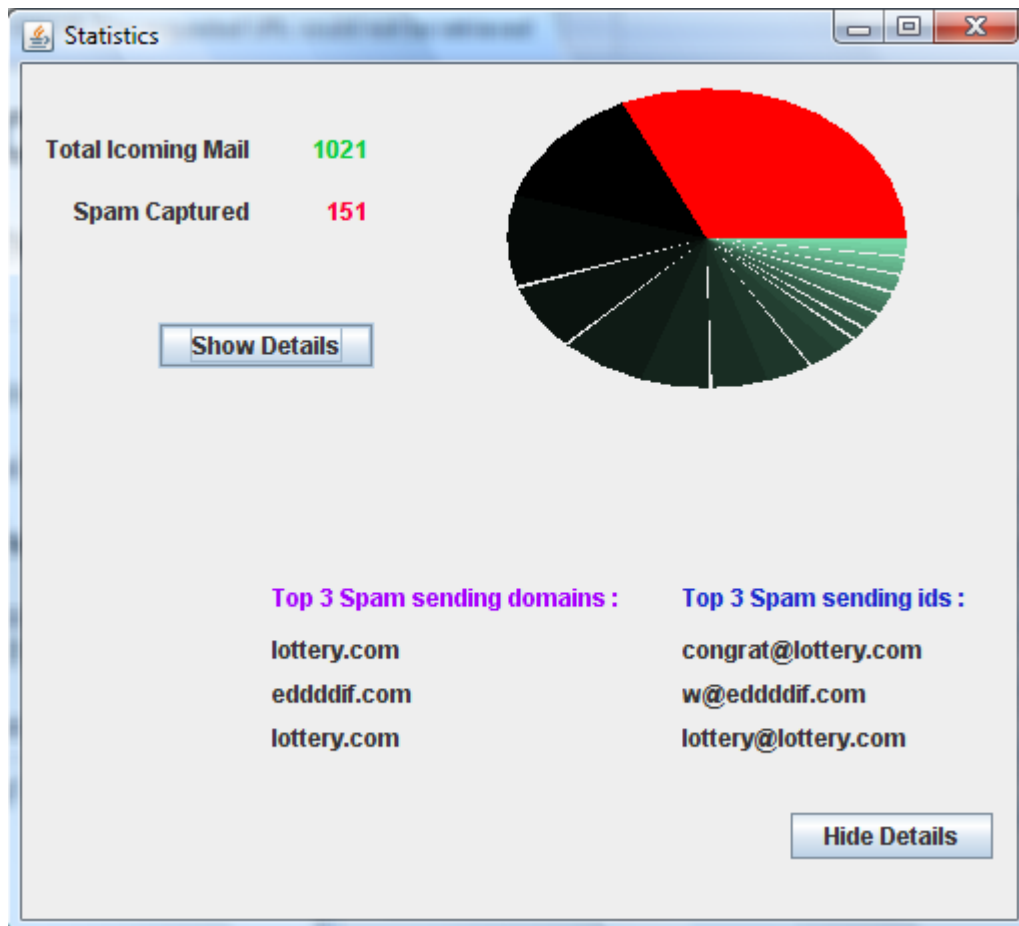


Figure 4

Clicking on "Show Details" button displays the top 5 spamming domains and top 5 spammer email addresses. A pie chart of spamming domain is also displayed.

Clicking on “Hide Details” hides the domains and spammer data.

## **Configuration Form**

The configuration of SpamWall is done in the form displayed in Figure 4.

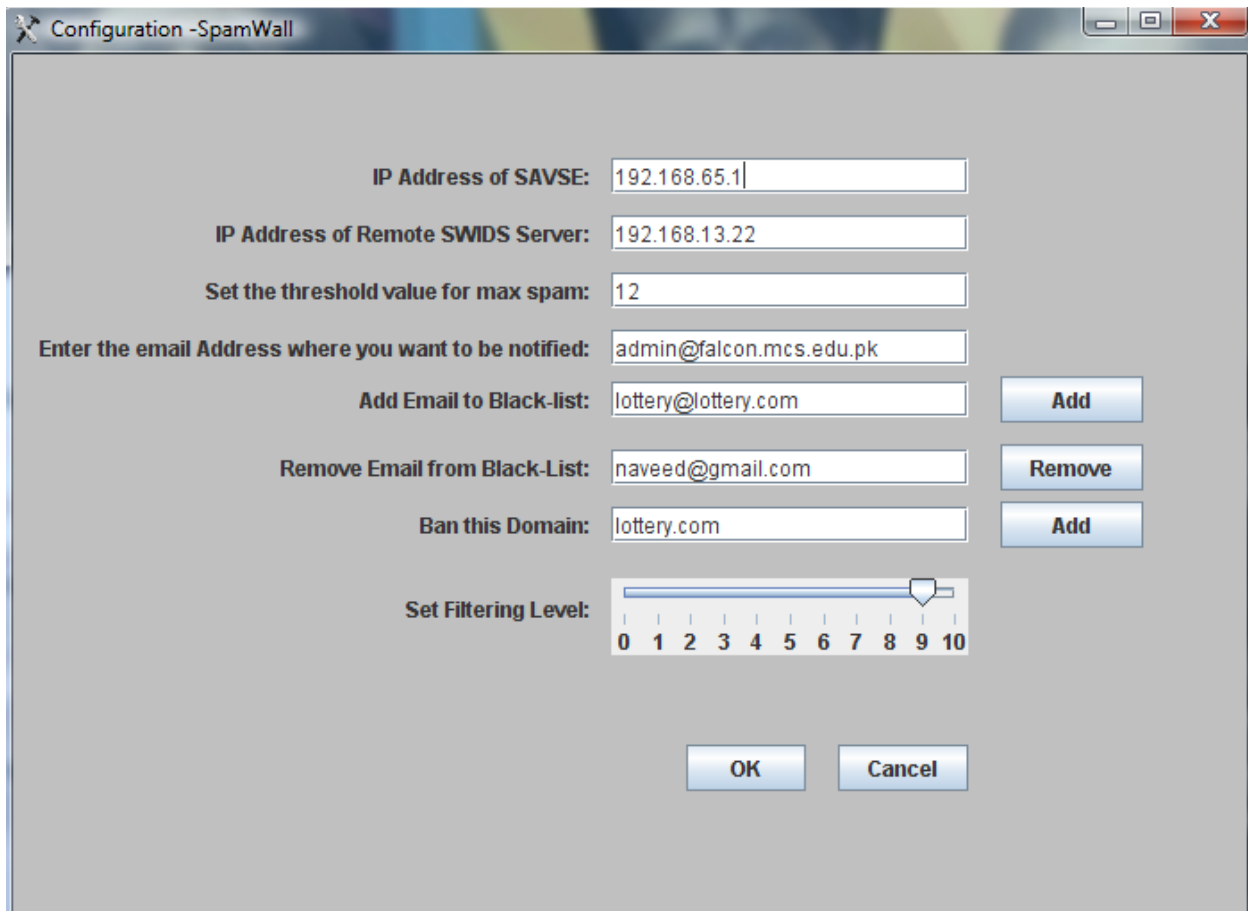


Figure 5

Administrator can do several configurations using this form.

He can set filtering level using Slider given in the form.

He can Blacklist a user or domain.

He can remove user from Blacklist.

He can set the maximum number of messages a spammer is tolerated. After that s system generated mail will be sent to Domain User defined by administrator.

## Tray Control

To increase the user friendliness SpamWall has its tray icon and a pop-up menu is displayed as depicted in Figure 5.

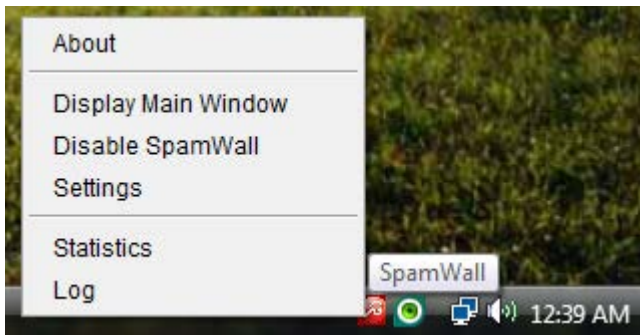


Figure 6

## **Bibliography**

[1] [http://en.wikipedia.org/wiki/E-mail\\_spam](http://en.wikipedia.org/wiki/E-mail_spam)

[2] <http://spam.abuse.net/overview/whatisspam.shtml>

[3] <http://www.paulgraham.com/spam.html>

[4] <http://www.viruslist.com>

[5] <http://java.sun.com>

[6] <http://www.javaexchangeconnector.com>

[7] <http://www.spam-site.com/anti-spam-techniques.shtml>

[8] <http://www.allspammedup.com/anti-spam/spam-filter-blacklists-and-whitelists>

、  
[9] [http://www.process.com/precisemail/bayesian\\_filtering.htm](http://www.process.com/precisemail/bayesian_filtering.htm)

[10] <http://www.research.ibm.com/spam/filtering.html>