

**A**

**Project Report**

**On**

*“Assessment For Security And Design Classifier”*

**Submitted in partial fulfillment of the requirements for the award of the degree of**

**Master of Science (Information System)**

**By**

**Ms. Shaymaa Taha Ahmed  
( ROLL NO:1009-13-863-058 )**

**Under the guidance of  
Mr. T. RAMDAS NAIK  
Asst. Professor**



**Department of Computer Science**

**NIZAM COLLEGE (AUTONOMOUS)**

**(A Constituent College, O.U)**

**BASHEER BAGH, Hyderabad**

**2014-2015**



**Department of Computer Science (PG)**  
**NIZAM COLLEGE (AUTONOMOUS)**  
(A Constituent College, O.U)

**CERTIFICATE**

This is to certify that the project entitled “**Assessment For Security And Design Classifier**” has been submitted by “**Ms. Shaymaa Taha Ahmed**” bearing Roll No: “**1009-13-863-058**” in the partial fulfilment of the requirements for the award of the degree of **Master of Science (Information System)** in Faculty of Informatics, Dept. of Computer Science, Nizam college, Osmania University, Hyderabad.

**PROJECT GUIDE**

**Mr. T. RAMDAS NAIK**

BE, MCA, M. Tech, (Ph. D)

**Assistant Professor**

**EXTERNAL**

**EXAMINER**

**Mr. T. RAMDAS NAIK**

BE, MCA, M. Tech, (Ph. D)

**Head of Department,  
Dept. of Informatics**

## **ACKNOWLEDGEMENT**

The success and final outcome of this project required a lot of guidance and assistance from many people and I am extremely fortunate to have got this all along the completion of my project work. Whatever I have done is only due to such guidance and assistance and I would not forget to thank them.

I owe my profound gratitude to my project guide **Mr. T. Ramdas Naik**, Asst. Professor and other teachers **Ms. S.Sravanthi** Asst. Professor, **Ms. Humera Shaziya** Asst. Professor, **Mr. Shaik Tanveer Ahmed**, Asst. Professor **Mr. O. Subhash Chander Goud**, Asst. Professor, who took keen interest on our project work and guided us all along, till the completion of our project work by providing all the necessary information for developing a good system.

I thank to **Mr. T. Ramdas Naik**, Head of the Dept. of Computer Science (PG), Nizam College, for his moral Support and Guidance.

I express my whole hearted gratitude to **Prof. T.L.N Swamy**, **Principal, Nizam College**, and to **Management Nizam College** for providing us the conducive environment to carry through our academic schedules and project with ease.

I am thankful to and fortunate enough to get constant encouragement, support and guidance from all Teaching staffs of Department of computer science which helped us in successfully completing our project work. Also, I would like to extend our sincere regards to all the non-teaching staff of department of computer science for their timely support.

Last but not the least; I would like to thank all my friends for their compliment.

Shaymaa Taha Ahmed  
Roll No: 1009-13-863-058

## **ABSTRACT**

**Title: “Assessment for Security and Design Classifier”**

With day by day influence of computerized system, the demand of automatic classification and filtering information come to the picture, Pattern classifier systems are one of those systems which speed up the process of filtering and classification of data. Pattern classification systems are commonly used in adversarial applications, like biometric authentication, network intrusion detection, and spam filtering, in which data can be purposely manipulated by humans to undermine their operation in effective and efficient way. As this adversarial scenario is not taken into account by classical design methods, pattern classification systems may exhibit vulnerabilities, whose exploitation may severely affect their performance, and consequently limit their practical utility.

Extending pattern classification theory and design methods to adversarial settings is thus a novel and very relevant research direction, which has not yet been pursued in a systematic way. In this paper, we address one of the main open issues: evaluating at design phase the security of pattern classifiers, namely, the performance degradation under potential attacks they may incur during operation. We propose a framework for empirical evaluation of classifier security that formalizes and generalizes the main ideas proposed in the literature, and give examples of its use in three real applications. Reported results show that security evaluation can provide a more complete understanding of the classifier’s behaviour in adversarial environments, and lead to better design choices.

# Table of Contents

<u>Contents</u>	<u>Page No.</u>
<b>CHAPTER ONE</b> -----	<b>1</b>
<b>1 Introduction</b> -----	<b>2</b>
1.1 Data Mining-----	5
1.1.1 What is Data Mining?-----	5
1.1.2 How Data Mining Works?-----	6
1.1.3. Advantages of Data Mining-----	10
1.2 Existing System-----	12
1.2.1 Disadvantages of Existing System-----	13
1.3 Proposed System -----	13
1.3.1 Advantages of Proposed System-----	14
1.4 Literature Survey -----	15
1.4.1 Robustness of Multimodal Biometric Fusion Methods against Spoof Attacks-----	15
1.4.2 Multimodal Fusion Vulnerability to Non-Zero Effort (Spoof) Imposters -----	15
1.4.3 Polymorphic Blending Attacks-----	16
1.4.4 On Attacking Statistical Spam Filters-----	17
1.4.5 Good Word Attacks on Statistical Spam Filters-----	18
<b>CHAPTER TWO</b> -----	<b>19</b>
<b>2 Feasibility Study</b> -----	<b>20</b>
2.1 Economical Feasibility-----	20
2.2 Technical Feasibility-----	21
2.3 Social Feasibility -----	22
<b>CHAPTER THREE</b> -----	<b>23</b>
<b>3 System Analyses</b> -----	<b>24</b>
3.1 Input Design -----	24
3.1.1 Objectives-----	25
3.2 Output Design -----	26
3.3 System Requirements -----	27
3.3.1 Hardware Requirements-----	27
3.3.2 Software Requirements-----	27
3.4 Technology Used -----	28
3.4.1 Java Technology-----	28
3.4.2 MySQL-----	58

**CHAPTER FOUR----- 62**

**4 System Design -----63**

- 4.1 System Architecture -----63
- 4.2 Data Flow Diagram -----64
- 4.3 UML Diagram -----67
  - 4.3.1 Goals -----68
  - 4.3.2 Use Case Diagram -----68
  - 4.3.3 Class Diagram -----70
  - 4.3.4 Sequence Diagram -----71
  - 4.3.5 Activity Duagram -----72

**CHAPTER FIVE ----- 73**

**5 System Implementation -----74**

- 5.1 Implementations -----74
  - 5.1.1 Modules -----74
  - 5.1.2 Modules' Descriptions -----74
    - 5.1.2.1 Attack Scenario and Model of the Adversary -----74
    - 5.1.2.2 Pattern Classification -----75
    - 5.1.2.3 Adversarial classification -----75
    - 5.1.2.4 Security modules -----76
- 5.2 Important Outputs Implementation -----77
- 5.3 Sample of source code -----88
  - 5.3.1 Admin Login action -----88
  - 5.3.2 User Registration Action -----89
  - 5.3.3 User Page -----93
  - 5.3.4 Spam Filter -----98

**CHAPTER SIX ----- 107**

**6 System Testing ----- 108**

- 6.1 Testing ----- 108
- 6.2 Types of Test ----- 108
  - 6.2.1 Unit Test ----- 108
  - 6.2.2 Integration Test ----- 109
  - 6.2.3 Functional Test ----- 109
  - 6.2.4 System Test ----- 110
  - 6.2.5 White Box Testing ----- 111
  - 6.2.6 Black Box Testing ----- 111

<b>CHAPTER SEVEN</b> -----	<b>114</b>
<b>Result</b> -----	<b>115</b>
<b><i>INDEXES</i></b> -----	<b>117</b>
<b><i>BIBLIOGRAPHY</i></b> -----	<b>118</b>

## **Table of Figures**

<b><u>Figure No.</u></b>	<b><u>Page No.</u></b>
Figure 1.1 Transformation of Data to knowledge.....	5
Figure 3.1 JAVA Compiler Steps.....	29
Figure 3.2 JVM Function.....	30
Figure 3.3 JAVA Platform.....	31
Figure 3.4 JAVA IDE.....	34
Figure 3.5 Steps Of JAVA program Compilation.....	43
Figure 3.6 TCP/IP model.....	44
Figure 3.7 IP address.....	47
Figure 3.8 Web Architecture of JAVA.....	57
Figure 4.1 System Architecture.....	64
Figure 4.2 Data Flow Diagram.....	66
Figure 4.3 Use Case Diagram.....	69
Figure 4.4 Class Diagram.....	70
Figure 4.5 Sequence Diagram.....	71
Figure 4.6 Activity Diagram.....	72
Figure 5.1 Home Screen.....	77
Figure 5.2 User Registration.....	78
Figure 5.3 Admin Login .....	79
Figure 5.4 Admin Welcome Page.....	80
Figure 5.5 Display of User List.....	81



Figure 5.6 Email Page.....	82
Figure 5.7 User Welcome Page.....	83
Figure 5.8 Public and Private Cloud File Upload.....	84
Figure 5.9 Display content of A File.....	85
Figure 5.10 Warning for Industry.....	86
Figure 5.11 List of Suspicious Attempters.....	87

**Chapter One**  
**Introduction**

## **1 Introduction**

**Assessment For Security And Design Classifier** is an evaluation work on pattern classifier systems, Pattern Classifiers based on machine learning algorithms are commonly used in security-related applications like biometric authentication, network intrusion detection, and spam filtering, to discriminate between a “legitimate” and a “malicious” pattern class (e.g., legitimate and spam emails).

Contrary to traditional ones, these applications have an intrinsic adversarial nature since the input data can be purposely manipulated by an intelligent and adaptive adversary to undermine classifier operation. This often gives rise to an arms race between the adversary and the classifier designer. Well known examples of attacks against pattern classifiers are: submitting a fake biometric trait to a biometric authentication system (spoofing attack); modifying network packets belonging to intrusive traffic to evade intrusion detection systems ; manipulating the content of spam emails to get them past spam filters (e.g., by misspelling common spam words to avoid their detection). Adversarial scenarios can also occur in intelligent data analysis and information retrieval; e.g., a malicious webmaster may manipulate search engine rankings to artificially promote her web site.

It is now acknowledged that, since pattern classification systems based on classical theory and design methods do not take into account

adversarial settings, they exhibit vulnerabilities to several potential attacks allowing adversaries to undermine their effectiveness.

A systematic and unified treatment of this issue is thus needed to allow the trusted adoption of pattern classifiers in adversarial environments, starting from the theoretical foundations up to novel design methods, extending the classical design cycle of. In particular, three main open issues can be identified:

- (i) Analyzing the vulnerabilities of classification algorithms, and the corresponding attacks.
- (ii) Developing novel methods to assess classifier security against these attacks, which are not possible using classical performance evaluation methods.
- (iii) Developing novel design methods to guarantee classifier security in adversarial environments.

Although this emerging field is attracting growing interest, the above issues have only been sparsely addressed under different perspectives and to a limited extent. Most of the work has focused on application-specific issues related to spam filtering and network intrusion detection, e.g, while only a few theoretical models of adversarial classification problems have been proposed in the machine learning literature; however, they do not yet provide practical guidelines and tools for designers of pattern recognition systems. Besides introducing these issues to the pattern recognition research community, in this work we address issues (i) and (ii) above by developing a framework for the empirical evaluation of classifier security at design phase that extends the

model selection and performance evaluation steps of the classical design cycle.

We summarize our work and point out three main ideas that emerge from it. We then formalize and generalize them in our framework.

First, to pursue security in the context of an arms race it is not sufficient to react to observed attacks, but it is also necessary to proactively anticipate the adversary by predicting the most relevant, potential attacks through a what-if analysis; this allows one to develop suitable countermeasures before the attack actually occurs, according to the principle of security by design.

Second, to provide practical guidelines for simulating realistic attack scenarios, we define a general model of the adversary, in terms of her goal, knowledge, and capability, which encompass and generalize models proposed in previous work.

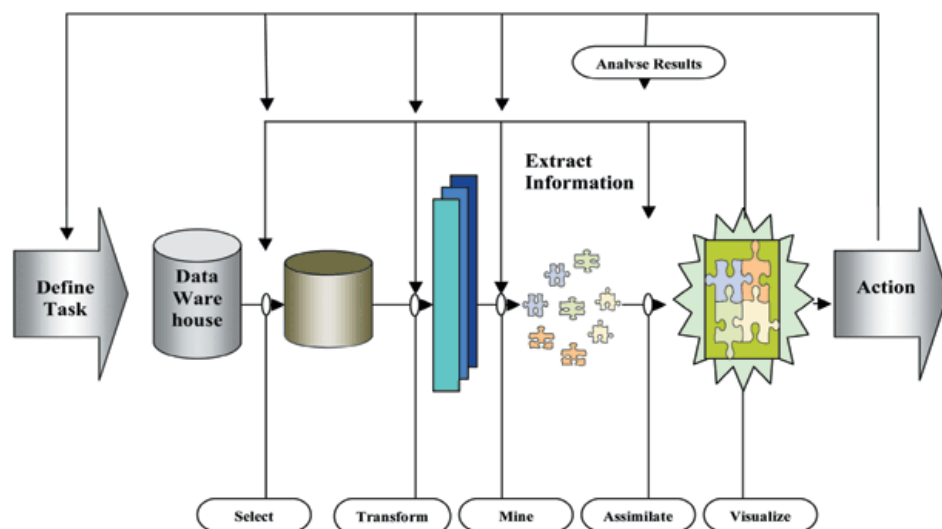
Third, since the presence of carefully targeted attacks may affect the distribution of training and testing data separately, we propose a model of the data distribution that can formally characterize this behaviour, and that allows us to take into account a large number of potential attacks; we also propose an algorithm for the generation of training and testing sets to be used for security evaluation, which can naturally accommodate application-specific and heuristic techniques for simulating attacks.

We give three concrete examples of applications of our framework in spam filtering, biometric authentication, and network intrusion detection. We discuss how the classical design cycle of pattern classifiers should be revised to take security into account

## 1.1 Data Mining

### 1.1.1 What is Data Mining?

Generally, data mining (sometimes called data or knowledge discovery) is the process of analyzing data from different perspectives and summarizing it into useful information - information that can be used to increase revenue, cuts costs, or both.



(Figure 1.1 Transformations of Data to Knowledge)

Data mining software is one of a number of analytical tools for analyzing data. It allows users to analyze data from many different dimensions or angles, categorize it, and summarize the relationships identified. Technically, data mining is the process of finding correlations or patterns among dozens of fields in large relational databases.

### **1.1.2 How Data Mining Works?**

While large-scale information technology has been evolving separate transaction and analytical systems, data mining provides the link between the two. Data mining software analyzes relationships and patterns in stored transaction data based on open-ended user queries. Several types of analytical software are available: statistical, machine learning, and neural networks.

Generally, any of four types of relationships are sought:

**Classes:** Stored data is used to locate data in predetermined groups. For example, a restaurant chain could mine customer purchase data to determine when customers visit and what they typically order. This information could be used to increase traffic by having daily specials.

**Clusters:** Data items are grouped according to logical relationships or consumer preferences. For example, data can be mined to identify market segments or consumer affinities.

**Associations:** Data can be mined to identify associations. The beer-diaper example is an example of associative mining.

**Sequential patterns:** Data is mined to anticipate behaviour patterns and trends. For example, an outdoor equipment retailer could predict the likelihood of a backpack being purchased based on a consumer's purchase of sleeping bags and hiking shoes.

### **Data mining consists of five major elements**

1. Extract, transform, and load transaction data onto the data warehouse system.
2. Store and manage the data in a multidimensional database system.
3. Provide data access to business analysts and information technology professionals.
4. Analyze the data by application software. Present the data in a useful format, such as a graph or table.

### **Different levels of analysis are available**

**Artificial neural networks:** Non-linear predictive models that learn through training and resemble biological neural networks in structure.



**Genetic algorithms:** Optimization techniques that use process such as genetic combination, mutation, and natural selection in a design based on the concepts of natural evolution.

**Decision trees:** Tree-shaped structures that represent sets of decisions. These decisions generate rules for the classification of a dataset. Specific decision tree methods include Classification and Regression Trees (CART) and Chi Square Automatic Interaction Detection (CHAID). CART and CHAID are decision tree techniques used for classification of a dataset. They provide a set of rules that you can apply to a new (unclassified) dataset to predict which records will have a given outcome. CART segments a dataset by creating 2-way splits while CHAID segments using chi square tests to create multi-way splits. CART typically requires less data preparation than CHAID.

**Nearest neighbour method:** A technique that classifies each record in a dataset based on a combination of the classes of the  $k$  record(s) most similar to it in a historical dataset (where  $k=1$ ). Sometimes called the k-nearest neighbor technique.

**Rule induction:** The extraction of useful if-then rules from data based on statistical significance.

**Data visualization:** The visual interpretation of complex relationships in multidimensional data. Graphics tools are used to illustrate data relationships.

## **Characteristics of Data Mining**

**Large quantities of data:** The volume of data so great it has to be analyzed by automated techniques e.g. satellite information, credit card transactions etc.

**Noisy, incomplete data:** Imprecise data is the characteristic of all data collection.

**Complex data structure:** conventional statistical analysis not possible. Heterogeneous data stored in legacy systems.

## **Benefits of Data Mining**

It's one of the most effective services that are available today. With the help of data mining, one can discover precious information about the customers and their behavior for a specific set of products and evaluate and analyze, store, mine and load data related to them.

An analytical CRM model and strategic business related decisions can be made with the help of data mining as it helps in providing a complete synopsis of customers

- 1) An endless number of organizations have installed data mining projects and it has helped them see their own companies make an unprecedented improvement in their marketing strategies (Campaigns)
- 2) Data mining is generally used by organizations with a solid customer focus. For its flexible nature as far as applicability is

concerned is being used vehemently in applications to foresee crucial data including industry analysis and consumer buying behaviors

- 3) Fast paced and prompt access to data along with economic processing techniques have made data mining one of the most suitable services that a company seek

### **1.1.3. Advantages of Data Mining**

#### **Marketing / Retail**

Data mining helps marketing companies build models based on historical data to predict who will respond to the new marketing campaigns such as direct mail, online marketing campaign...etc. Through the results, marketers will have appropriate approach to sell profitable products to targeted customers.

Data mining brings a lot of benefits to retail companies in the same way as marketing. Through market basket analysis, a store can have an appropriate production arrangement in a way that customers can buy frequent buying products together with pleasant. In addition, it also helps the retail companies offer certain discounts for particular products that will attract more customers.

**Finance / Banking**

Data mining gives financial institutions information about loan information and credit reporting. By building a model from historical customer's data, the bank and financial institution can determine good and bad loans. In addition, data mining helps banks detect fraudulent credit card transactions to protect credit card's owner.

**Manufacturing**

By applying data mining in operational engineering data, manufacturers can detect faulty equipments and determine optimal control parameters. For example semi-conductor manufacturers has a challenge that even the conditions of manufacturing environments at different wafer production plants are similar, the quality of wafer are lot the same and some for unknown reasons even has defects. Data mining has been applying to determine the ranges of control parameters that lead to the production of golden wafer. Then those optimal control parameters are used to manufacture wafers with desired quality.

**Governments**

Data mining helps government agency by digging and analyzing records of financial transaction to build patterns that can detect money laundering or criminal activities.

**Law enforcement**

Data mining can aid law enforcers in identifying criminal suspects as well as apprehending these criminals by examining trends in location, crime type, habit, and other patterns of behaviours.

### **Researchers**

Data mining can assist researchers by speeding up their data analyzing process; thus, allowing those more time to work on other projects.

## **1.2 Existing System**

Pattern classification systems based on classical theory and design methods do not take into account adversarial settings; they exhibit vulnerabilities to several potential attacks, allowing adversaries to undermine their effectiveness. A systematic and unified treatment of this issue is thus needed to allow the trusted adoption of pattern classifiers in adversarial environments, starting from the theoretical foundations up to novel design methods, extending the classical design cycle of. In particular, three main open issues can be identified:

- (i) Analyze the vulnerabilities of classification algorithms, and the corresponding attacks.

- (ii) Developing novel methods to assess classifier security against these attacks, which are not possible using classical performance evaluation methods.
- (iii) Developing novel design methods to guarantee classifier security in adversarial environments.

### **1.2.1 Disadvantages of Existing System**

1. Poor analyzing the vulnerabilities of classification algorithms, and the corresponding attacks.
2. A malicious webmaster may manipulate search engine rankings to artificially promote website.

### **1.3 Proposed System**

In this work we address issues above by developing a framework for the empirical evaluation of classifier security at design phase that extends the model selection and performance evaluation steps of the classical design cycle .We summarize previous work, and point out three main ideas that emerge from it. We then formalize and generalize them in our framework. First, to pursue security in the context of an arms race it is not sufficient to react to observed attacks, but it is also necessary to proactively anticipate the adversary by predicting the most relevant,

potential attacks through a what-if analysis; this allows one to develop suitable countermeasures before the attack actually occurs, according to the principle of security by design. Second, to provide practical guidelines for simulating realistic attack scenarios, we define a general model of the adversary, in terms of her goal, knowledge, and capability, which encompass and generalize models proposed in previous work. Third, since the presence of carefully targeted attacks may affect the distribution of training and testing data separately, we propose a model of the data distribution that can formally characterize this behaviour, and that allows us to take into account a large number of potential attacks; we also propose an algorithm for the generation of training and testing sets to be used for security evaluation, which can naturally accommodate application-specific and heuristic techniques for simulating attacks.

### **1.3.1 Advantages of Proposed System**

1. Proposed system prevents developing novel methods to assess classifier security against these attacks.
2. The presence of an intelligent and adaptive adversary makes the classification problem highly non-stationary.

## 1.4 Literature Survey

### 1.4.1 Robustness of Multimodal Biometric Fusion Methods against Spoof Attacks

**AUTHORS:** R.N. Rodrigues, L.L. Ling, and V. Govindaraju

In this paper, we address the security of multimodal biometric systems when one of the modes is successfully spoofed. We propose two novel fusion schemes that can increase the security of multimodal biometric systems. The first is an extension of the likelihood ratio based fusion scheme and the other uses fuzzy logic. Besides the matching score and sample quality score, our proposed fusion schemes also take into account the intrinsic security of each biometric system being fused. Experimental results have shown that the proposed methods are more robust against spoof attacks when compared with traditional fusion methods.

### 1.4.2 Multimodal Fusion Vulnerability to Non-Zero Effort (Spoof) Imposters

**AUTHORS:** P. Johnson, B. Tan, and S. Schuckers

In biometric systems, the threat of “spoofing”, where an imposter will fake a biometric trait, has led to the increased use of multimodal biometric systems. It is assumed that an imposter must spoof all



modalities in the system to be accepted. This paper looks at the cases where some but not all modalities are spoofed. The contribution of this paper is to outline a method for assessment of multimodal systems and underlying fusion algorithms. The framework for this method is described and experiments are conducted on a multimodal database of face, iris, and fingerprint match scores.

### **1.4.3 Polymorphic Blending Attacks**

**AUTHORS:** P. Fogla, M. Sharif, R. Perdisci, O. Kolesnikov, and W. Lee

A very effective means to evade signature-based intrusion detection systems (IDS) is to employ polymorphic techniques to generate attack instances that do not share a fixed signature.

Anomaly-based intrusion detection systems provide good defense because existing polymorphic techniques can make the attack instances look different from each other, but cannot make them look like normal. In this paper we introduce a new class of polymorphic attacks, called polymorphic blending attacks, that can effectively evade byte frequency-based network anomaly IDS by carefully matching the statistics of the mutated attack instances to the normal profiles. The proposed polymorphic blending attacks can be viewed as a subclass of the mimicry attacks.

We take a systematic approach to the problem and formally describe the algorithms and steps required to carry out such attacks. We

not only show that such attacks are feasible but also analyze the hardness of evasion under different circumstances. We present detailed techniques using PAYL, a byte frequency-based anomaly IDS, as a case study and demonstrate that these attacks are indeed feasible. We also provide some insight into possible countermeasures that can be used as defence.

#### **1.4.4 On Attacking Statistical Spam Filters**

**AUTHORS:** G.L. Wittel and S.F. Wu

The efforts of anti-spammers and spammers have often been described as an arms race. As we devise new ways to stem the flood of bulk mail, spammers respond by working their way around the new mechanisms. Their attempts to bypass spam filters illustrate this struggle. Spammers have tried many things from using HTML layout tricks, letter substitution, to adding random data. While at times their attacks are clever, they have yet to work strongly against the statistical nature that drives many filtering systems.

The challenges in successfully developing such an attack are great as the variety of filtering systems makes it less likely that a single attack can work against all of them. Here, we examine the general attack methods spammers use, along with challenges faced by developers and spammers. We also demonstrate an attack that, while easy to implement, attempts to more strongly work against the statistical nature behind filters.

### **1.4.5 Good Word Attacks on Statistical Spam Filters**

**AUTHORS:** D. Lowd and C. Meek

Unsolicited commercial email is a significant problem for users and providers of email services. While statistical spam filters have proven useful, senders of spam are learning to bypass these filters by systematically modifying their email messages. In a good word attack, one of the most common techniques, a spammer modifies a spam message by inserting or appending words indicative of legitimate email.

. In this paper, we describe and evaluate the effectiveness of active and passive good word attacks against two types of statistical spam filters: naive Bayes and maximum entropy filters. We find that in passive attacks without any filter feedback, an attacker can get 50 % of currently blocked spam past either filter by adding 150 words or fewer. In active attacks allowing test queries to the target filter, 30 words will get half of blocked spam past either filter.

**Chapter Two**  
**Feasibility Study**

## **2 Feasibility Study**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ◆ ECONOMICAL FEASIBILITY
- ◆ TECHNICAL FEASIBILITY
- ◆ SOCIAL FEASIBILITY

### **2.1 Economical Feasibility**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the

technologies used are freely available. Only the customized products had to be purchased.

The **Assessment for Security and Design Classifier** is requiring resources which are either freely available or can be purchased with a minimum affordable budget almost for many companies and organizations, thus we can say that implementation of this applications is economically feasible.

## **2.2 Technical Feasibility**

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

The required hardware and software for the implementation of the **Assessment for Security and Design Classifier** application is available anywhere with very cheap cost and nowadays almost many organizations even small businesses are using the same H/W and S/W for different purposes, thus the **Assessment for Security and Design Classifier** is technically feasible to be implemented for majority of organizations.

### 2.3 Social Feasibility

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

As nowadays most of organizations' employees are familiar to use data mining tools like BI etc. due to similarity of **Assessment for Security and Design Classifier** with other pattern classification systems applications the users are able to understand the software easily and at the same time it has extra features like adding a new fake words as far as the policy is changed so it provides a kind of flexibility to the system. At the same time the screens are guided with user friendly guidelines thus we can say that the Assessment for Security and Design Classifier is socially feasible to be implemented.

**Chapter Three**  
**System Analysis**



## 3 System Analyses

### 3.1 Input Design

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

### **3.1.1 Objectives**

1. Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow.

### **3.2 Output Design**

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important

and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

2. Select methods for presenting information.

3. Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives:

- ❖ Convey information about past activities, current status or projections of the
- ❖ Future.
- ❖ Signal important events, opportunities, problems, or warnings.
- ❖ Trigger an action.
- ❖ Confirm an action.

### **3.3 System Requirements**

#### **3.3.1 Hardware Requirements**

- System : Pentium IV 2.4 GHz.
- Hard Disk : 40 GB.
- Floppy Drive : 1.44 Mb.
- Monitor : 15 VGA Colour.
- Mouse : Logitech.
- Ram : 512 Mb.

#### **3.3.2 Software Requirements**

- Operating system : Windows XP/7.
- Coding Language : JAVA/J2EE
- IDE : Netbeans 7.4
- Database : MYSQL

### **3.4 Technology Used**

#### **3.4.1 Java Technology**

Java technology is both a programming language and a platform.

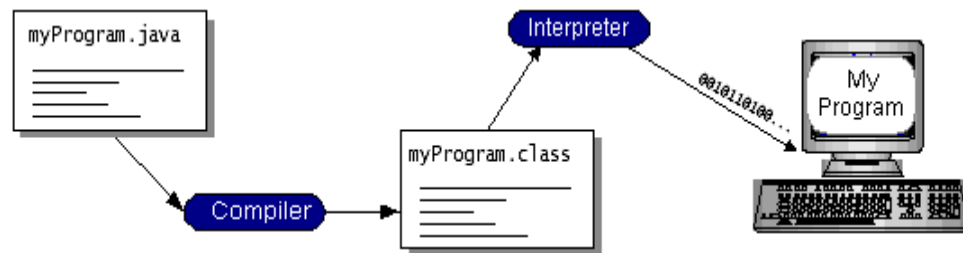
#### **The Java Programming Language**

The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

- Simple
- Architecture neutral
- Object oriented
- Portable
- Distributed
- High performance
- Interpreted
- Multithreaded
- Robust
- Dynamic
- Secure

With most programming languages, you either compile or interpret a program so that you can run it on your computer. The Java

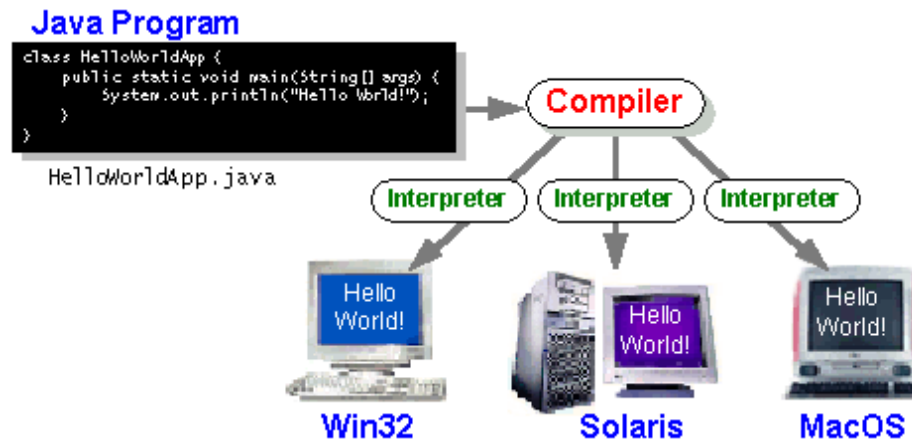
programming language is unusual in that a program is both compiled and interpreted. With the compiler, first you translate a program into an intermediate language called *Java byte codes* —the platform-independent codes interpreted by the interpreter on the Java platform. The interpreter parses and runs each Java byte code instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed. The following figure illustrates how this works.



(Figure 3.1 JAVA compiler steps)

You can think of Java byte codes as the machine code instructions for the Java Virtual Machine (Java VM). Every Java interpreter, whether it's a development tool or a Web browser that can run applets, is an implementation of the Java VM. Java byte codes help make “write once, run anywhere” possible. You can compile your program into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the Java VM. That means that as long as a computer has a Java VM, the same program written in the Java

programming language can run on Windows 2000, a Solaris workstation, or on an iMac.



(Figure 3.2 JVM Functions)

## The Java Platform

A platform is the hardware or software environment in which a program runs. We've already mentioned some of the most popular platforms like Windows 2000, Linux, Solaris, and MacOS. Most platforms can be described as a combination of the operating system and hardware. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms.

The Java platform has two components:

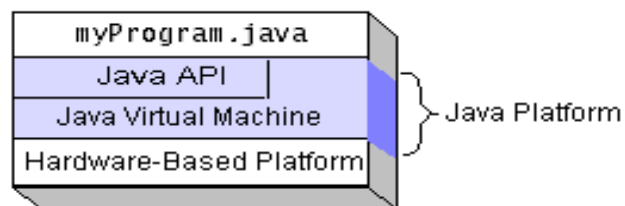
### The Java Virtual Machine (Java VM)

### The Java Application Programming Interface (Java API)

You've already been introduced to the Java VM. It's the base for the Java platform and is ported onto various hardware-based platforms.

The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped into libraries of related classes and interfaces; these libraries are known as packages. The next section, *What Can Java Technology Do? Highlights* what functionality some of the packages in the Java API provide.

The following figure depicts a program that's running on the Java platform. As the figure shows, the Java API and the virtual machine insulate the program from the hardware.



( Figure 3.3 JAVA platform)



Native code is code that after you compile it, the compiled code runs on a specific hardware platform. As a platform-independent environment, the Java platform can be a bit slower than native code. However, smart compilers, well-tuned interpreters, and just-in-time byte code compilers can bring performance close to that of native code without threatening portability.

### **What Can Java Technology Do?**

The most common types of programs written in the Java programming language are *applets* and *applications*. If you've surfed the Web, you're probably already familiar with applets. An applet is a program that adheres to certain conventions that allow it to run within a Java-enabled browser.

However, the Java programming language is not just for writing cute, entertaining applets for the Web. The general-purpose, high-level Java programming language is also a powerful software platform. Using the generous API, you can write many types of programs.

An application is a standalone program that runs directly on the Java platform. A special kind of application known as a server serves and supports clients on a network. Examples of servers are Web servers, proxy servers, mail servers, and print servers. Another specialized program is a servlet. A servlet can almost be thought of as an applet that runs on the server side. Java Servlets are a popular choice for building interactive web applications, replacing the use of CGI scripts. Servlets are similar to applets in that they are runtime extensions of applications.

Instead of working in browsers, though, servlets run within Java Web servers, configuring or tailoring the server.

How does the API support all these kinds of programs? It does so with packages of software components that provides a wide range of functionality. Every full implementation of the Java platform gives you the following features:

**The essentials:** Objects, strings, threads, numbers, input and output, data structures, system properties, date and time, and so on.

**Applets:** The set of conventions used by applets.

**Networking:** URLs, TCP (Transmission Control Protocol), UDP (User Data gram Protocol) sockets, and IP (Internet Protocol) addresses.

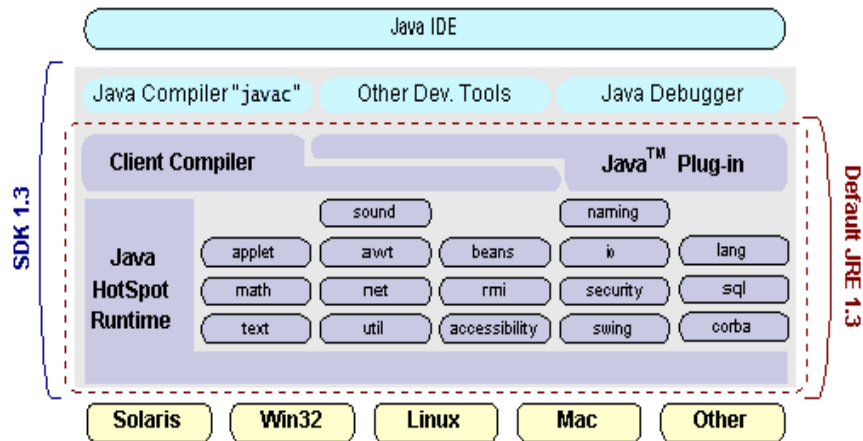
**Internationalization:** Help for writing programs that can be localized for users worldwide. Programs can automatically adapt to specific locales and be displayed in the appropriate language.

**Security:** Both low level and high level, including electronic signatures, public and private key management, access control, and certificates.

**Software components:** Known as JavaBeans<sup>™</sup>, can plug into existing component architectures.

**Object serialization:** Allows lightweight persistence and communication via Remote Method Invocation (RMI).

**Java Database Connectivity (JDBC<sup>™</sup>):** Provides uniform access to a wide range of relational databases.



(Figure 3.4 JAVA IDE)

The Java platform also has APIs for 2D and 3D graphics, accessibility servers, collaboration, telephony, speech, animation, and more. The following figure depicts what is included in the Java 2 SDK..

### How Will Java Technology Change My Life?

We can't promise you fame, fortune, or even a job if you learn the Java programming language. Still, it is likely to make your programs better and requires less effort than other languages. We believe that Java technology will help you do the following:

**Get started quickly:** Although the Java programming language is a powerful object-oriented language, it's easy to learn, especially for programmers already familiar with C or C++.

**Write less code:** Comparisons of program metrics (class counts, method counts, and so on) suggest that a program written in the Java programming language can be four times smaller than the same program in C++.

**Write better code:** The Java programming language encourages good coding practices, and its garbage collection helps you avoid memory leaks. Its object orientation, its JavaBeans component architecture, and its wide-ranging, easily extendible API let you reuse other people's tested code and introduce fewer bugs.

**Develop programs more quickly:** Your development time may be as much as twice as fast versus writing the same program in C++. Why? You write fewer lines of code and it is a simpler programming language than C++.

**Avoid platform dependencies with 100% Pure Java:** You can keep your program portable by avoiding the use of libraries written in other languages. The 100% Pure Java™ Product Certification Program has a repository of historical process manuals, white papers, brochures, and similar materials online.

**Write once, run anywhere:** Because 100% Pure Java programs are compiled into machine-independent byte codes, they run consistently on any Java platform.

**Distribute software more easily:** You can upgrade applets easily from a central server. Applets take advantage of the feature of allowing new classes to be loaded “on the fly,” without recompiling the entire program.

## **ODBC**

Microsoft Open Database Connectivity (ODBC) is a standard programming interface for application developers and database systems providers. Before ODBC became a de facto standard for Windows programs to interface with database systems, programmers had to use proprietary languages for each database they wanted to connect to. Now, ODBC has made the choice of the database system almost irrelevant from a coding perspective, which is as it should be. Application developers have much more important things to worry about than the syntax that is needed to port their program from one database to another when business needs suddenly change.

Through the ODBC Administrator in Control Panel, you can specify the particular database that is associated with a data source that an ODBC application program is written to use. Think of an ODBC data source as a door with a name on it. Each door will lead you to a particular database. For example, the data source named Sales Figures might be a

SQL Server database, whereas the Accounts Payable data source could refer to an Access database. The physical database referred to by a data source can reside anywhere on the LAN.

The ODBC system files are not installed on your system by Windows 95. Rather, they are installed when you setup a separate database application, such as SQL Server Client or Visual Basic 4.0. When the ODBC icon is installed in Control Panel, it uses a file called ODBCINST.DLL. It is also possible to administer your ODBC data sources through a stand-alone program called ODBCADM.EXE. There is a 16-bit and a 32-bit version of this program and each maintains a separate list of ODBC data sources.

From a programming perspective, the beauty of ODBC is that the application can be written to use the same set of function calls to interface with any data source, regardless of the database vendor. The source code of the application doesn't change whether it talks to Oracle or SQL Server. We only mention these two as an example. There are ODBC drivers available for several dozen popular database systems. Even Excel spreadsheets and plain text files can be turned into data sources. The operating system uses the Registry information written by ODBC Administrator to determine which low-level ODBC drivers are needed to talk to the data source (such as the interface to Oracle or SQL Server). The loading of the ODBC drivers is transparent to the ODBC application program. In a client/server environment, the ODBC API even handles many of the network issues for the application programmer.

The advantages of this scheme are so numerous that you are probably thinking there must be some catch. The only disadvantage of ODBC is that it isn't as efficient as talking directly to the native database interface. ODBC has had many detractors make the charge that it is too slow. Microsoft has always claimed that the critical factor in performance is the quality of the driver software that is used. In our humble opinion, this is true. The availability of good ODBC drivers has improved a great deal recently. And anyway, the criticism about performance is somewhat analogous to those who said that compilers would never match the speed of pure assembly language. Maybe not, but the compiler (or ODBC) gives you the opportunity to write cleaner programs, which means you finish sooner. Meanwhile, computers get faster every year.

## **JDBC**

In an effort to set an independent database standard API for Java; Sun Microsystems developed Java Database Connectivity, or JDBC. JDBC offers a generic SQL database access mechanism that provides a consistent interface to a variety of RDBMSs. This consistent interface is achieved through the use of "plug-in" database connectivity modules, or drivers. If a database vendor wishes to have JDBC support, he or she must provide the driver for each platform that the database and Java run on.

To gain a wider acceptance of JDBC, Sun based JDBC's framework on ODBC. As you discovered earlier in this chapter, ODBC has widespread support on a variety of platforms. Basing JDBC on ODBC will allow vendors to bring JDBC drivers to market much faster than developing a completely new connectivity solution.

JDBC was announced in March of 1996. It was released for a 90 day public review that ended June 8, 1996. Because of user input, the final JDBC v1.0 specification was released soon after.

The remainder of this section will cover enough information about JDBC for you to know what it is about and how to use it effectively. This is by no means a complete overview of JDBC. That would fill an entire book.

### **JDBC Goals**

Few software packages are designed without goals in mind. JDBC is one that, because of its many goals, drove the development of the API. These goals, in conjunction with early reviewer feedback, have finalized the JDBC class library into a solid framework for building database applications in Java.

The goals that were set for JDBC are important. They will give you some insight as to why certain classes and functionalities behave the way they do. The eight design goals for JDBC are as follows:



## **1.SQL Level API**

The designers felt that their main goal was to define a SQL interface for Java. Although not the lowest database interface level possible, it is at a low enough level for higher-level tools and APIs to be created. Conversely, it is at a high enough level for application programmers to use it confidently. Attaining this goal allows for future tool vendors to “generate” JDBC code and to hide many of JDBC’s complexities from the end user.

## **2.SQL Conformance**

SQL syntax varies as you move from database vendor to database vendor. In an effort to support a wide variety of vendors, JDBC will allow any query statement to be passed through it to the underlying database driver. This allows the connectivity module to handle non-standard functionality in a manner that is suitable for its users.

JDBC must be implemental on top of common database interfaces  
The JDBC SQL API must “sit” on top of other common SQL level APIs. This goal allows JDBC to use existing ODBC level drivers by the use of a software interface. This interface would translate JDBC calls to ODBC and vice versa.

Provide a Java interface that is consistent with the rest of the Java system

Because of Java's acceptance in the user community thus far, the designers feel that they should not stray from the current design of the core Java system.

### **Keep it simple**

This goal probably appears in all software design goal listings. JDBC is no exception. Sun felt that the design of JDBC should be very simple, allowing for only one method of completing a task per mechanism. Allowing duplicate functionality only serves to confuse the users of the API.

### **Use strong, static typing wherever possible**

Strong typing allows for more error checking to be done at compile time; also, less error appear at runtime.

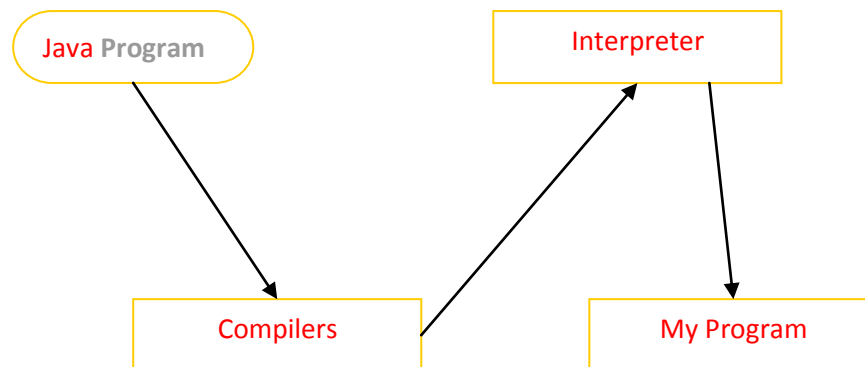
### **Keep the common cases simple**

Because more often than not, the usual SQL calls used by the programmer are simple SELECT's, INSERT's, DELETE's and UPDATE's, these queries should be simple to perform with JDBC. However, more complex SQL statements should also be possible.

Java has two things: a programming language and a platform. Java is a high-level programming language that is all of the following

Simple	Architecture-neutral
Object-oriented	Portable
Distributed	High-performance
Interpreted	multithreaded
Robust	Dynamic
Secure	

Java is also unusual in that each Java program is both compiled and interpreted. With a compile you translate a Java program into an intermediate language called Java byte codes the platform-independent code instruction is passed and run on the computer. Compilation happens just once; interpretation occurs each time the program is executed. The figure illustrates how this works.



(Figure 3.5 Steps Of JAVA Program Compilation)

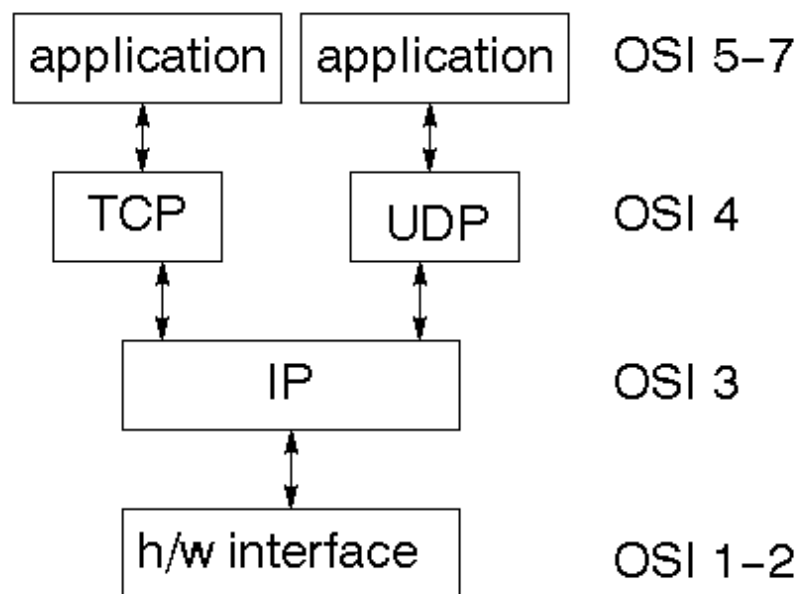
You can think of Java byte codes as the machine code instructions for the Java Virtual Machine (Java VM). Every Java interpreter, whether it's a Java development tool or a Web browser that can run Java applets, is an implementation of the Java VM. The Java VM can also be implemented in hardware.

Java byte codes help make “write once, run anywhere” possible. You can compile your Java program into byte codes on my platform that has a Java compiler. The byte codes can then be run any implementation of the Java VM. For example, the same Java program can run Windows NT, Solaris, and Macintosh.

## Networking

### TCP/IP stack

The TCP/IP stack is shorter than the OSI one:



(Figure 3.6 TCP/IP model)

TCP is a connection-oriented protocol; UDP (User Datagram Protocol) is a connectionless protocol.

**IP datagram's**

The IP layer provides a connectionless and unreliable delivery system. It considers each datagram independently of the others. Any association between datagram must be supplied by the higher layers. The IP layer supplies a checksum that includes its own header. The header includes the source and destination addresses. The IP layer handles routing through an Internet. It is also responsible for breaking up large datagram into smaller ones for transmission and reassembling them at the other end.

**UDP**

UDP is also connectionless and unreliable. What it adds to IP is a checksum for the contents of the datagram and port numbers. These are used to give a client/server model - see later.

**TCP**

TCP supplies logic to give a reliable connection-oriented protocol above IP. It provides a virtual circuit that two processes can use to communicate.

**Internet addresses**

In order to use a service, you must be able to find it. The Internet uses an address scheme for machines so that they can be located. The address is a 32 bit integer which gives the IP address. This encodes a network ID and more addressing. The network ID falls into various classes according to the size of the network address.

**Network address**

Class A uses 8 bits for the network address with 24 bits left over for other addressing. Class B uses 16 bit network addressing. Class C uses 24 bit network addressing and class D uses all 32.

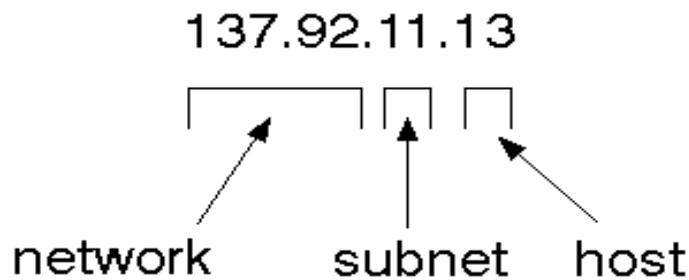
**Subnet address**

Internally, the UNIX network is divided into sub networks. Building 11 is currently on one sub network and uses 10-bit addressing, allowing 1024 different hosts.

**Host address**

8 bits are finally used for host addresses within our subnet. This places a limit of 256 machines that can be on the subnet.

## Total address



(Figure 3.7 IP address)

The 32 bit address is usually written as 4 integers separated by dots.

## Port addresses

A service exists on a host, and is identified by its port. This is a 16 bit number. To send a message to a server, you send it to the port for that service of the host that it is running on. This is not location transparency! Certain of these ports are "well known".

## Sockets

A socket is a data structure maintained by the system to handle network connections. A socket is created using the call `socket`. It returns an integer that is like a file descriptor. In fact, under Windows, this handle can be used with `Read File` and `Write File` functions.

```
#include <sys/types.h>
#include <sys/socket.h>

int socket(int family, int type, int protocol);
```



Here "family" will be `AF_INET` for IP communications, `protocol` will be zero, and `type` will depend on whether TCP or UDP is used. Two processes wishing to communicate over a network create a socket each. These are similar to two ends of a pipe - but the actual pipe does not yet exist.

### **JFree Chart**

JFreeChart is a free 100% Java chart library that makes it easy for developers to display professional quality charts in their applications. JFreeChart's extensive feature set includes:

A consistent and well-documented API, supporting a wide range of chart types;

A flexible design that is easy to extend, and targets both server-side and client-side applications;

Support for many output types, including Swing components, image files (including PNG and JPEG), and vector graphics file formats (including PDF, EPS and SVG);

JFreeChart is "open source" or, more specifically, free software. It is distributed under the terms of the GNU Lesser General Public Licence (LGPL), which permits use in proprietary applications.

## **1. Map Visualizations**

Charts showing values that relate to geographical areas. Some examples include: (a) population density in each state of the United States, (b) income per capita for each country in Europe, (c) life expectancy in each country of the world. The tasks in this project include:

Sourcing freely redistributable vector outlines for the countries of the world, states/provinces in particular countries (USA in particular, but also other areas); Creating an appropriate dataset interface (plus default implementation), a rendered, and integrating this with the existing XYPlot class in JFreeChart; Testing, documenting, testing some more, documenting some more.

## **2. Time Series Chart Interactivity**

Implement a new (to JFreeChart) feature for interactive time series charts --- to display a separate control that shows a small version of ALL the time series data, with a sliding "view" rectangle that allows you to select the subset of the time series data to display in the main chart.

## **3. Dashboards**

There is currently a lot of interest in dashboard displays. Create a flexible dashboard mechanism that supports a subset of JFreeChart chart

types (dials, pies, thermometers, bars, and lines/time series) that can be delivered easily via both Java Web Start and an applet.

#### **4. Property Editors**

The property editor mechanism in JFreeChart only handles a small subset of the properties that can be set for charts. Extend(or reimplement) this mechanism to provide greater end-user control over the appearance of the charts.

#### **What is a Java Web Application?**

A Java web application generates interactive web pages containing various types of markup language (HTML, XML, and so on) and dynamic content. It is typically comprised of web components such as JavaServer Pages (JSP), servlets and JavaBeans to modify and temporarily store data, interact with databases and web services, and render content in response to client requests.

Because many of the tasks involved in web application development can be repetitive or require a surplus of boilerplate code, web frameworks can be applied to alleviate the overhead associated with common activities. For example, many frameworks, such as JavaServer Faces, provide libraries for templating pages and session management, and often promote code reuse.

### **What is Java EE?**

Java EE (Enterprise Edition) is a widely used platform containing a set of coordinated technologies that significantly reduce the cost and complexity of developing, deploying, and managing multi-tier, server-centric applications. Java EE builds upon the Java SE platform and provides a set of APIs (application programming interfaces) for developing and running portable, robust, scalable, reliable and secure server-side applications.

Some of the fundamental components of Java EE include:

- Enterprise JavaBeans (EJB): a managed, server-side component architecture used to encapsulate the business logic of an application. EJB technology enables rapid and simplified development of distributed, transactional, secure and portable applications based on Java technology.
- Java Persistence API (JPA): a framework that allows developers to manage data using object-relational mapping (ORM) in applications built on the Java Platform.

### **JavaScript and Ajax Development**

JavaScript is an object-oriented scripting language primarily used in client-side interfaces for web applications. Ajax (Asynchronous JavaScript and XML) is a Web 2.0 technique that allows changes to occur in a web page without the need to perform a page refresh. JavaScript

toolkits can be leveraged to implement Ajax-enabled components and functionality in web pages.

### **Web Server and Client**

Web Server is a software that can process the client request and send the response back to the client. For example, Apache is one of the most widely used web server. Web Server runs on some physical machine and listens to client request on specific port.

A web client is a software that helps in communicating with the server. Some of the most widely used web clients are Firefox, Google Chrome, Safari etc. When we request something from server (through URL), web client takes care of creating a request and sending it to server and then parsing the server response and present it to the user.

### **HTML and HTTP**

Web Server and Web Client are two separate softwares, so there should be some common language for communication. HTML is the common language between server and client and stands for **H**yper**T**ext **M**arkup **L**anguage.

Web server and client needs a common communication protocol, HTTP (**H**yper**T**ext **T**ransfer **P**rotocol) is the communication protocol between server and client. HTTP runs on top of TCP/IP communication protocol.

Some of the important parts of HTTP Request are:

- **HTTP Method** – action to be performed, usually GET, POST, PUT etc.
- **URL** – Page to access
- **Form Parameters** – similar to arguments in a java method, for example user,password details from login page.

Sample HTTP Request:

- 1 GET /FirstServletProject/jsp/hello.jsp HTTP/1.1
- 2 Host: localhost:8080
- 3 Cache-Control: no-cache

Some of the important parts of HTTP Response are:

- **Status Code** – an integer to indicate whether the request was success or not. Some of the well known status codes are 200 for success, 404 for Not Found and 403 for Access Forbidden.
- **Content Type** – text, html, image, pdf etc. Also known as MIME type
- **Content** – actual data that is rendered by client and shown to user.

**MIME Type or Content Type:** If you see above sample HTTP response header, it contains tag “Content-Type”. It’s also called MIME type and server sends it to client to let them know the kind of data it’s sending. It helps client in rendering the data for user. Some of the mostly used mime types are text/html, text/xml, application/xml etc.

## Understanding URL

URL is acronym of Universal Resource Locator and it's used to locate the server and resource. Every resource on the web has it's own unique address. Let's see parts of URL with an example.

**http://localhost:8080/FirstServletProject/jsps/hello.jsp**

**http://** – This is the first part of URL and provides the communication protocol to be used in server-client communication.

**localhost** – The unique address of the server, most of the times it's the hostname of the server that maps to unique IP address. Sometimes multiple hostnames point to same IP addresses and web server virtual host takes care of sending request to the particular server instance.

**8080** – This is the port on which server is listening, it's optional and if we don't provide it in URL then request goes to the default port of the protocol. Port numbers 0 to 1023 are reserved ports for well known services, for example 80 for HTTP, 443 for HTTPS, 21 for FTP etc.

**FirstServletProject/jsps/hello.jsp** – Resource requested from server. It can be static html, pdf, JSP, servlets, PHP etc.

## Why we need Servlet and JSPs?

Web servers are good for static contents HTML pages but they don't know how to generate dynamic content or how to save data into databases, so we need another tool that we can use to generate dynamic

content. There are several programming languages for dynamic content like PHP, Python, Ruby on Rails, Java Servlets and JSPs.

Java Servlet and JSPs are server side technologies to extend the capability of web servers by providing support for dynamic response and data persistence.

### **Web Container**

Tomcat is a web container, when a request is made from Client to web server, it passes the request to web container and it's web container job to find the correct resource to handle the request (servlet or JSP) and then use the response from the resource to generate the response and provide it to web server. Then web server sends the response back to the client.

When web container gets the request and if it's for servlet then container creates two Objects HTTP Servlet Request and HttpServletResponse. Then it finds the correct servlet based on the URL and creates a thread for the request. Then it invokes the servlet service() method and based on the HTTP method service() method invokes doGet() or doPost() methods. Servlet methods generate the dynamic page and write it to response. Once servlet thread is complete, container converts the response to HTTP response and send it back to client.

Some of the important work done by web container are:

**Communication Support** – Container provides easy way of



communication between web server and the servlets and JSPs. Because of container, we don't need to build a server socket to listen for any request from web server, parse the request and generate response. All these important and complex tasks are done by container and all we need to focus is on our business logic for our applications.

**Lifecycle and Resource Management** – Container takes care of managing the life cycle of servlet. Container takes care of loading the servlets into memory, initializing servlets, invoking servlet methods and destroying them. Container also provides utility like JNDI for resource pooling and management.

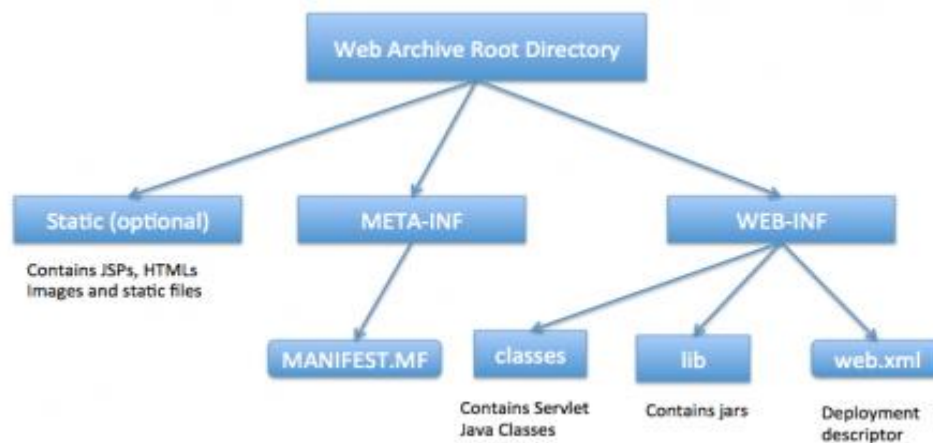
**Multithreading Support** – Container creates new thread for every request to the servlet and when it's processed the thread dies. So servlets are not initialized for each request and saves time and memory.

**JSP Support** – JSPs doesn't look like normal java classes and web container provides support for JSP. Every JSP in the application is compiled by container and converted to Servlet and then container manages them like other servlets.

**Miscellaneous Task** – Web container manages the resource pool, does memory optimizations, run garbage collector, provides security configurations, support for multiple applications, hot deployment and several other tasks behind the scene that makes our life easier.

## Web Application Directory Structure

Java Web Applications are packaged as Web Archive (WAR) and it has a defined structure. You can export above dynamic web project as WAR file and unzip it to check the hierarchy. It will be something like below image.



(Figure 3.8 Web App Architecture of JAVA)

## Deployment Descriptor

**web.xml** file is the deployment descriptor of the web application and contains mapping for servlets (prior to 3.0), welcome pages, security configurations, session timeout settings etc.

That's all for the Java web application startup tutorial, we will explore Servlets and JSPs more in future posts.

### **3.4.2 MySQL**

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation.

The MySQL Web site (<http://www.mysql.com/>) provides the latest information about MySQL software.

#### **MySQL is a database management system.**

A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.

#### **MySQL databases are relational.**

A relational database stores data in separate tables rather than putting all the data in one big storeroom. The database structures are organized into physical files optimized for speed. The logical model, with objects such as databases, tables, views, rows, and columns, offers a

flexible programming environment. You set up rules governing the relationships between different data fields, such as one-to-one, one-to-many, unique, required or optional, and “pointers” between different tables. The database enforces these rules, so that with a well-designed database, your application never sees inconsistent, duplicate, orphan, out-of-date, or missing data.

The SQL part of “MySQL” stands for “Structured Query Language”. SQL is the most common standardized language used to access databases. Depending on your programming environment, you might enter SQL directly (for example, to generate reports), embed SQL statements into code written in another language, or use a language-specific API that hides the SQL syntax.

SQL is defined by the ANSI/ISO SQL Standard. The SQL standard has been evolving since 1986 and several versions exist. In this manual, “SQL-92” refers to the standard released in 1992, “SQL:1999” refers to the standard released in 1999, and “SQL:2003” refers to the current version of the standard. We use the phrase “the SQL standard” to mean the current version of the SQL Standard at any time.

### **MySQL software is Open Source.**

Open Source means that it is possible for anyone to use and modify the software. Anybody can download the MySQL software from the Internet and use it without paying anything. If you wish, you may study the source code and change it to suit your needs. The MySQL software uses the GPL (GNU General Public License), <http://www.fsf.org/licenses/>, to define what you may and may not do with

the software in different situations. If you feel uncomfortable with the GPL or need to embed MySQL code into a commercial application, you can buy a commercially licensed version from us. See the MySQL Licensing Overview for more information (<http://www.mysql.com/company/legal/licensing/>).

**The MySQL Database Server is very fast, reliable, scalable, and easy to use.**

If that is what you are looking for, you should give it a try. MySQL Server can run comfortably on a desktop or laptop, alongside your other applications, web servers, and so on, requiring little or no attention. If you dedicate an entire machine to MySQL, you can adjust the settings to take advantage of all the memory, CPU power, and I/O capacity available. MySQL can also scale up to clusters of machines, networked together.

You can find a performance comparison of MySQL Server with other database managers on our benchmark page.

MySQL Server was originally developed to handle large databases much faster than existing solutions and has been successfully used in highly demanding production environments for several years. Although under constant development, MySQL Server today offers a rich and useful set of functions. Its connectivity, speed, and security make MySQL Server highly suited for accessing databases on the Internet.

**MySQL Server works in client/server or embedded systems.**

The MySQL Database Software is a client/server system that consists of a multi-threaded SQL server that supports different backends, several different client programs and libraries, administrative tools, and a wide range of application programming interfaces (APIs).

We also provide MySQL Server as an embedded multi-threaded library that you can link into your application to get a smaller, faster, easier-to-manage standalone product.

**A large amount of contributed MySQL software is available.**

MySQL Server has a practical set of features developed in close cooperation with our users. It is very likely that your favorite application or language supports the MySQL Database Server.

The official way to pronounce “MySQL” is “My Ess Que Ell” (not “my sequel”), but we do not mind if you pronounce it as “my sequel” or in some other localized way.

**Chapter Four**  
**System Design**

## 4 System Design

### 4.1 System Architecture

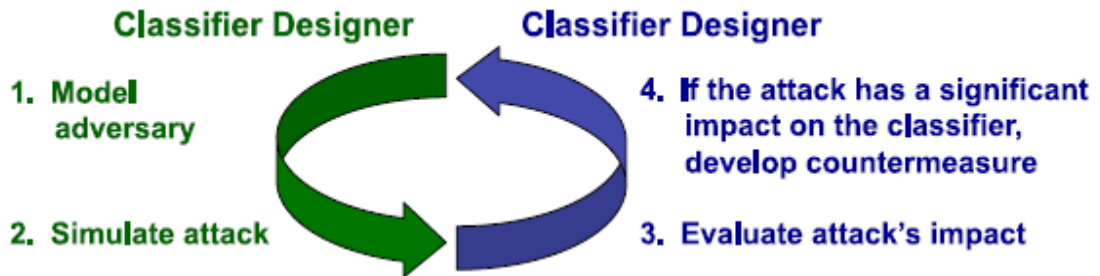
To secure a system, a common approach used in engineering and cryptography is security by obscurity that relies on keeping secret some of the system details to the adversary. In contrast, the paradigm of security by design advocates that systems should be designed from the ground-up to be secure, without assuming that the adversary may ever find out some important system details. Accordingly, the system designer should anticipate the adversary by simulating a “proactive” arms race as show in the figure.

This paradigm typically improves security by delaying each step of the “reactive” arms race, as it requires the adversary to spend a greater effort (time, skills, and resources) to find and exploit vulnerabilities. System security should thus be guaranteed for a longer time, with less frequent supervision or human intervention.

The goal of security evaluation is to address issue:

To simulate a number of realistic attack scenarios that may be incurred during operation, and to assess the impact of the corresponding attacks on the targeted classifier to highlight the most critical vulnerabilities. This amounts to performing a what-if analysis, which is a





(Figure 4.1 System architecture)

common practice in security. Although security evaluation may also suggest specific countermeasures, the design of secure classifiers.

## 4.2 Data Flow Diagram

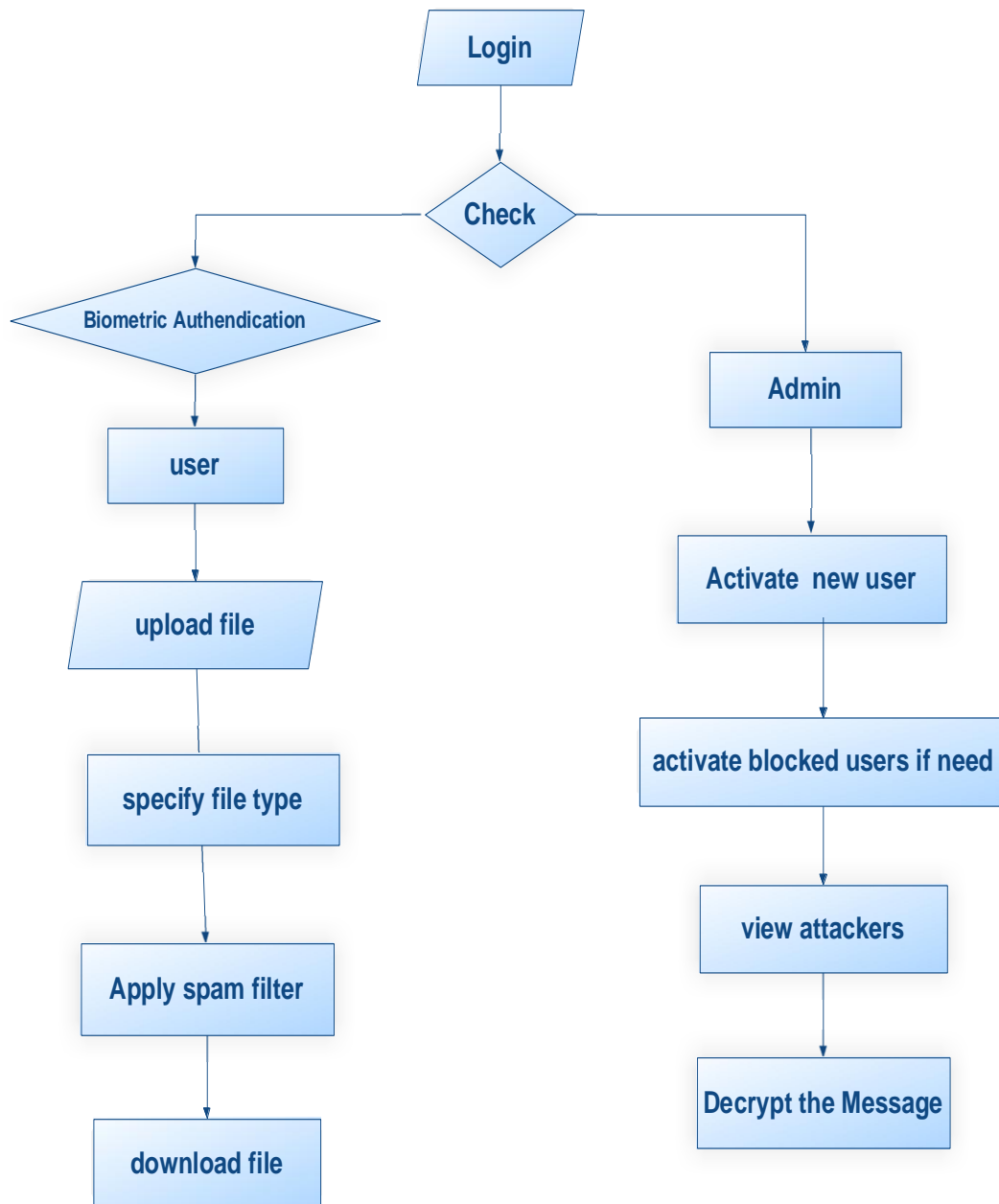
1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.

3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

The user and admin can login with the user ID and password, the admin has to activate the newly registered user then he is able to do activities.

The activities user can do is log in, upload file, specify file type, applying spam filter and downloading the file from the cloud.

The activities that admin can do are activate user, activate blocked user, view attacks and decrypt the messages.



(Figure 4.2 Data Flow Diagram)

### **4.3 UML Diagram**

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

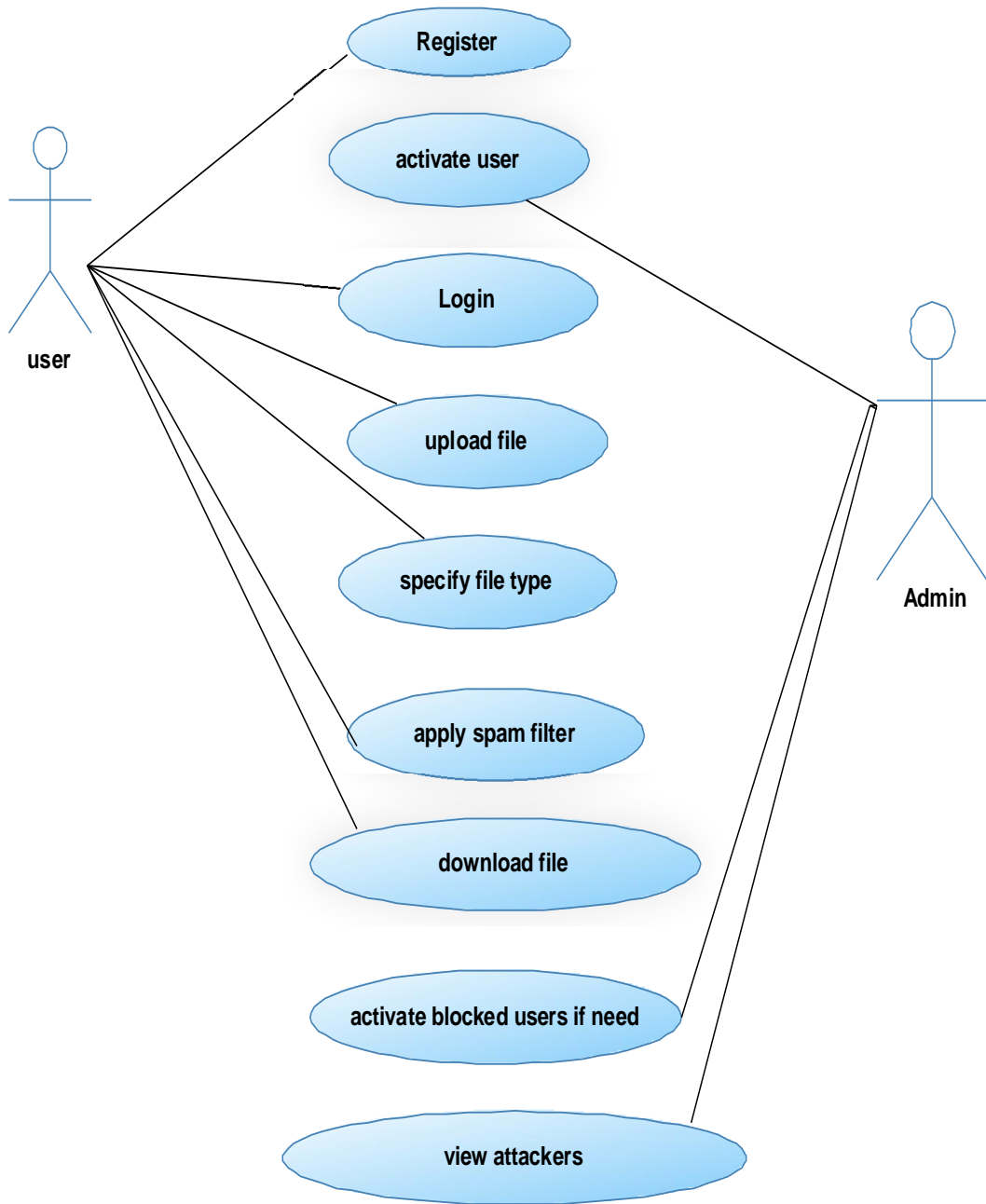
### **4.3.1 Goals**

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

### **4.3.2 Use Case Diagram**

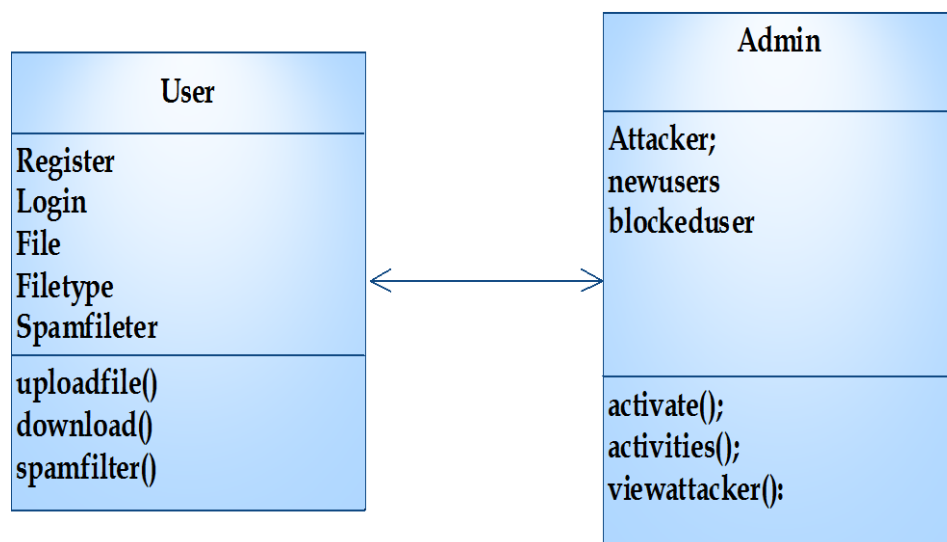
A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



(Figure 4.3 Use Case Diagram)

### 4.3.3 Class Diagram

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

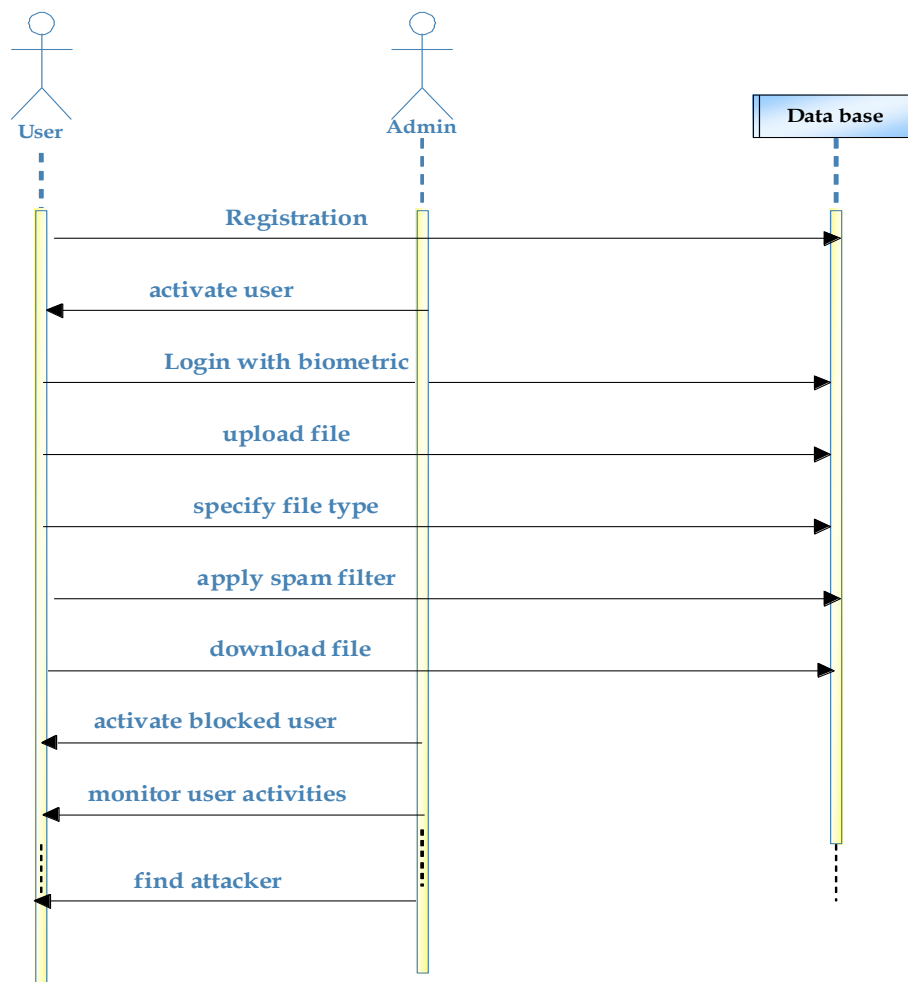


(Figure 4.4 Class Diagram)

The user methods are nothing but File upload, Selecting file type, Applying spam filter and downloading the files. Likewise the admin methods are Activating the user, Activating the blocked user and viewing attackers.

### 4.3.4 Sequence Diagram

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagram.

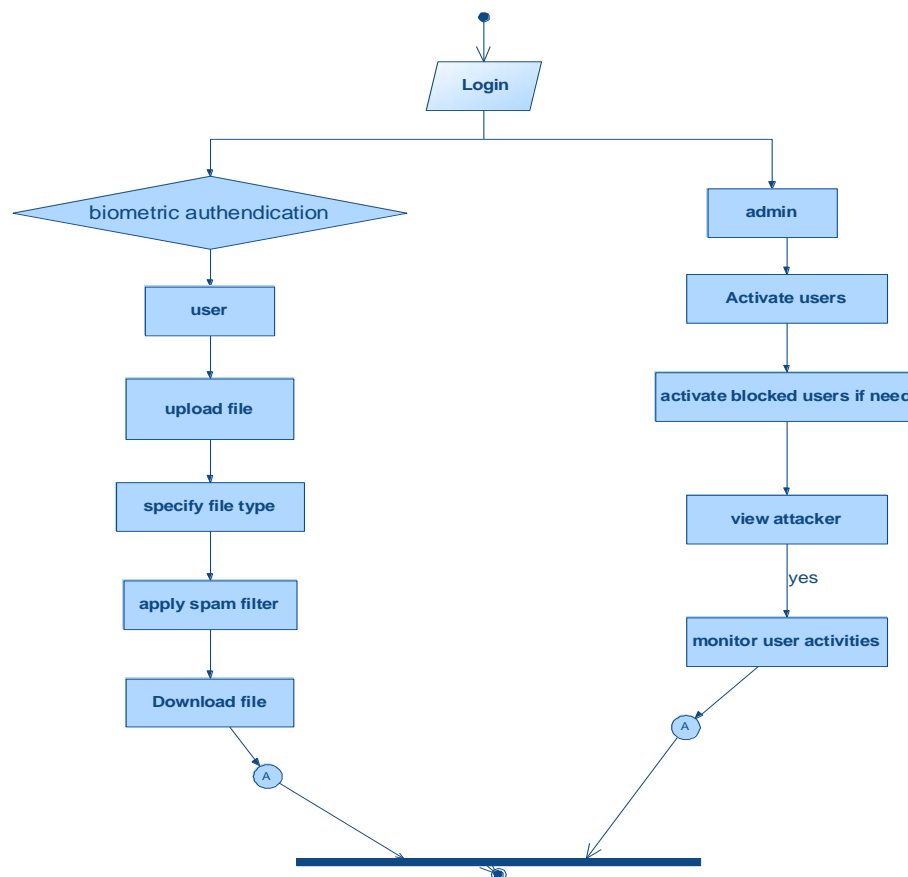


(Figure 4.5 Sequence Diagram)



### 4.3.5 Activity Duagram

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.



(Figure 4.6 Activity Diagram)

**Chapter Five**  
**System Implementation**  
**&**  
**Important Outputs**

## **5 System Implementation**

### **5.1 Implementations**

#### **5.1.1 Modules**

1. Attack Scenario and Model of the Adversary
2. Pattern Classification
3. Adversarial classification:
4. Security modules

#### **5.1.2 Modules' Descriptions**

##### **5.1.2.1 Attack Scenario and Model of the Adversary**

Although the definition of attack scenarios is ultimately an application-specific issue, it is possible to give general guidelines that can help the designer of a pattern recognition system. Here we propose to specify the attack scenario in terms of a conceptual model of the adversary that encompasses, unifies, and extends different ideas from previous work. Our model is based on the assumption that the adversary acts rationally to attain a given goal, according to her knowledge of the

classifier, and her capability of manipulating data. This allows one to derive the corresponding optimal attack strategy.

### **5.1.2.2 Pattern Classification**

Multimodal biometric systems for personal identity recognition have received great interest in the past few years. It has been shown that combining information coming from different biometric traits can overcome the limits and the weaknesses inherent in every individual biometric, resulting in a higher accuracy. Moreover, it is commonly believed that multimodal systems also improve security against Spoofing attacks, which consist of claiming a false identity and submitting at least one fake biometric trait to the system (e.g., a “gummy” fingerprint or a photograph of a user’s face). The reason is that, to evade multimodal system, one expects that the adversary should spoof all the corresponding biometric traits. In this application example, we show how the designer of a multimodal system can verify if this hypothesis holds, before deploying the system, by simulating spoofing attacks against each of the matchers.

### **5.1.2.3 Adversarial classification**

Assume that a classifier has to discriminate between legitimate and spam emails on the basis of their textual content, and that the bag-of-

words feature representation has been chosen, with binary features denoting the occurrence of a given set of words.

#### **5.1.2.4 Security modules**

Intrusion detection systems analyze network traffic to prevent and detect malicious activities like intrusion attempts, ROC curves of the considered multimodal biometric system under a simulated spoof attack against the fingerprint or the face matcher. Port scans, and denial-of-service attacks. When suspected malicious traffic is detected, an alarm is raised by the IDS and subsequently handled by the system administrator. Two main kinds of IDSs exist: misuse detectors and anomaly-based ones. Misuse detectors match the analyzed network traffic against a database of signatures of known malicious activities. The main drawback is that they are not able to detect never-before-seen malicious activities, or even variants of known ones. To overcome this issue, anomaly-based detectors have been proposed. They build a statistical model of the normal traffic using machine learning techniques, usually one-class classifiers, and raise an alarm when anomalous traffic is detected. Their training set is constructed, and periodically updated to follow the changes of normal traffic, by collecting unsupervised network traffic during operation, assuming that it is normal (it can be filtered by a misuse detector, and should)

## 5.2 Important Outputs Implementation

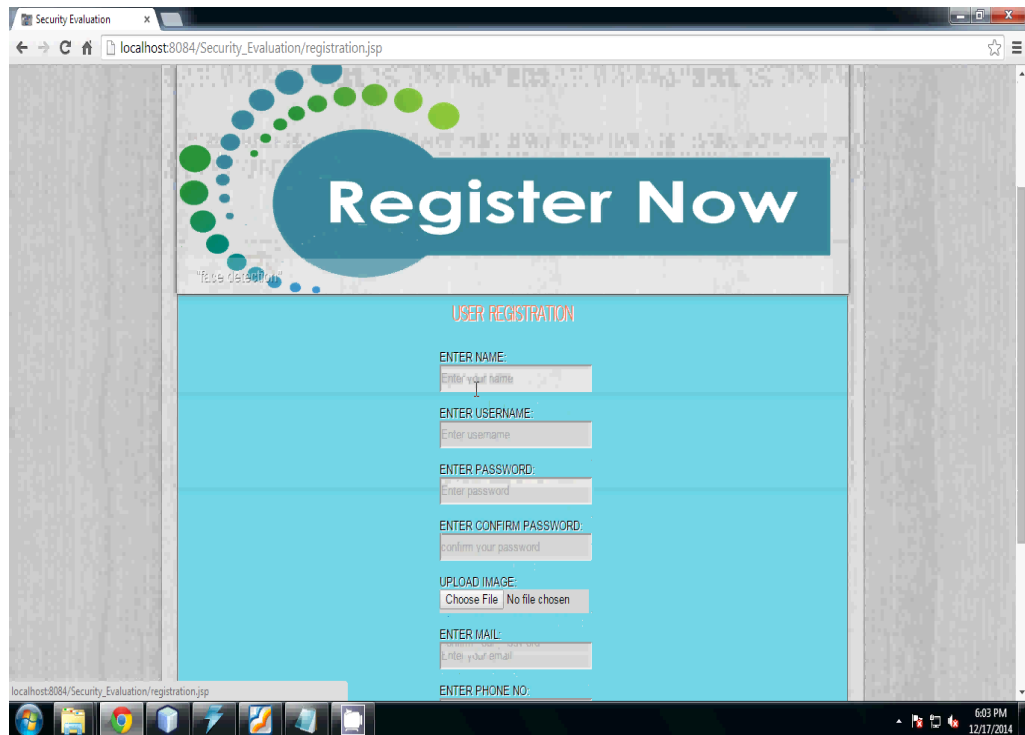
### Home Screen



(Figure 5.1 Home Screen)

The home screen allows user and admin to login and if a user is new he can register here also.

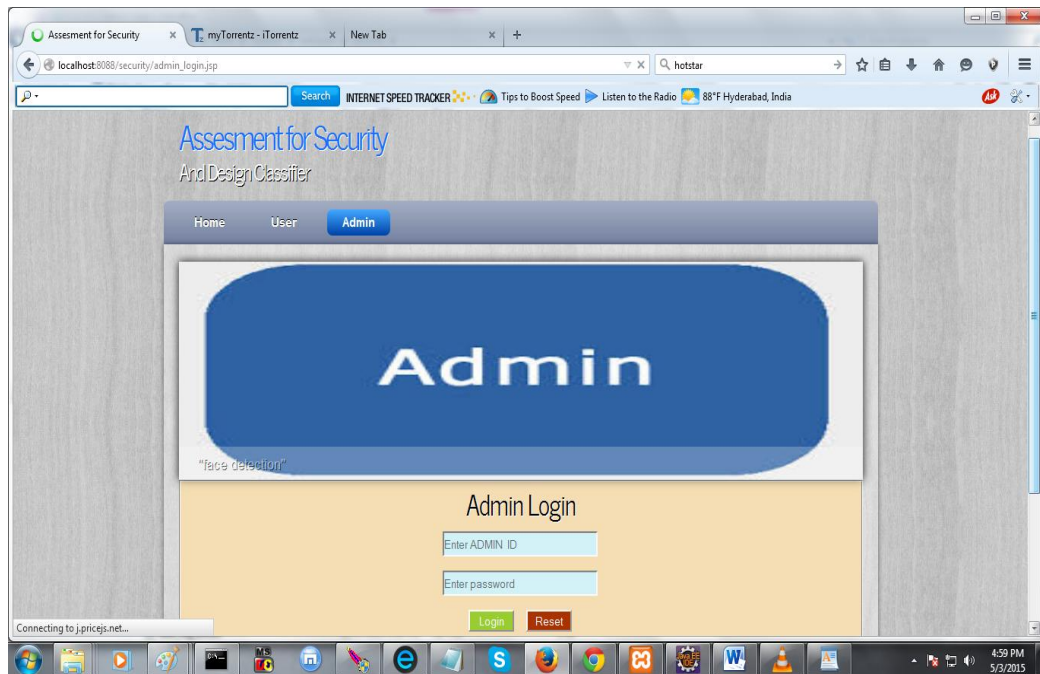
## User Registration Screen



(Figure 5.2 User Registration Screen)

In this page a user can register for first time. User has to enter all the informations required

## Admin Login page

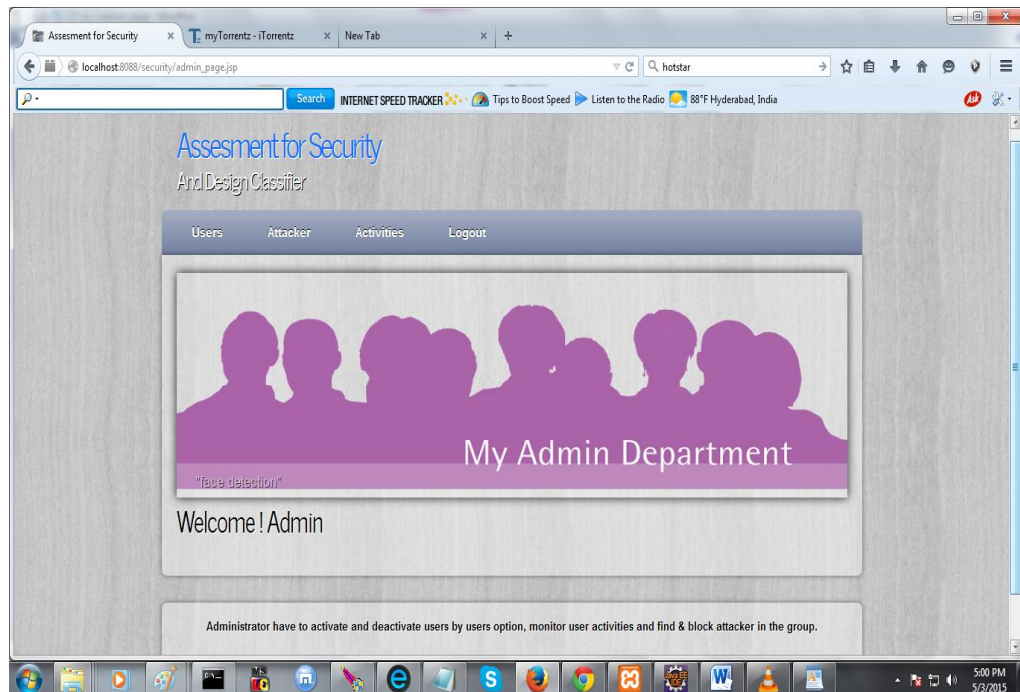


(Figure 5.3 Admin Login Page)

This Page Allow admin to login and activate the users for first time or reactivate the users who are blocked. Admin has full access to see the attempters or attackers who are tried to log in using invalid User Name and password for three times.



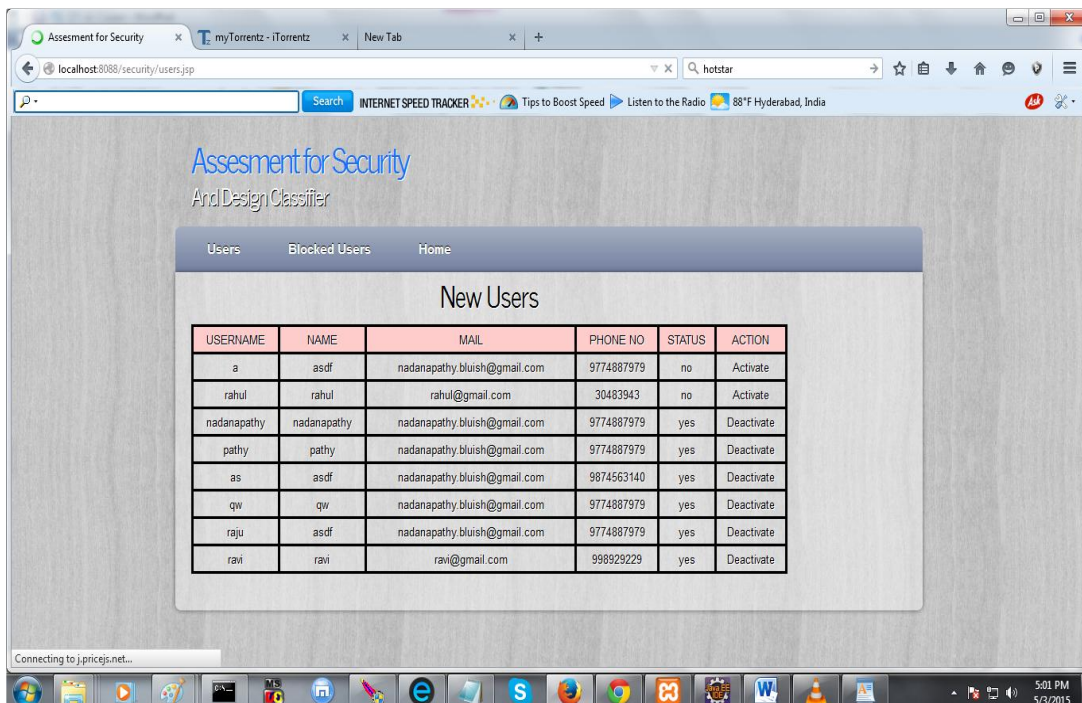
## Admin Welcome Page



(Figure 5.4 Admin Welcome Page)

This Page give access to admin to activate users, have a look to attackers and activities. Admin can see all the files which are smap filter applied and also non spma filter applied.

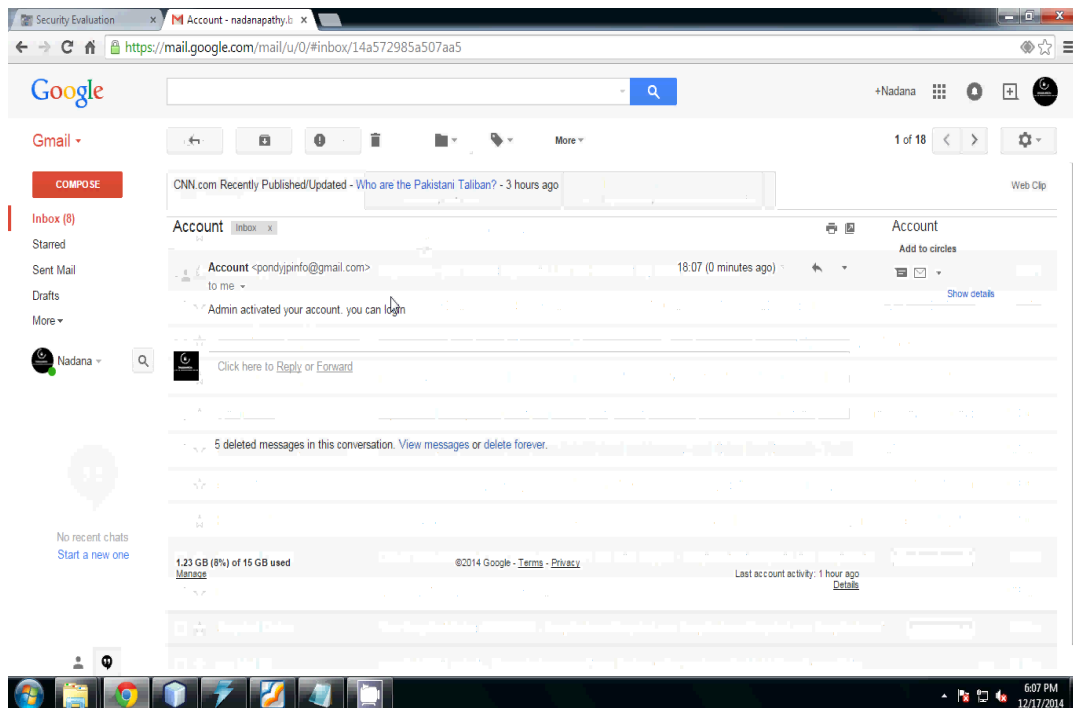
## User List Screen



(Figure 5.5 Display of Users List)

This Page shows users' details and the users' status whether a user is active or deactivates and give to admin the access to activate or deactivate an appropriate user.

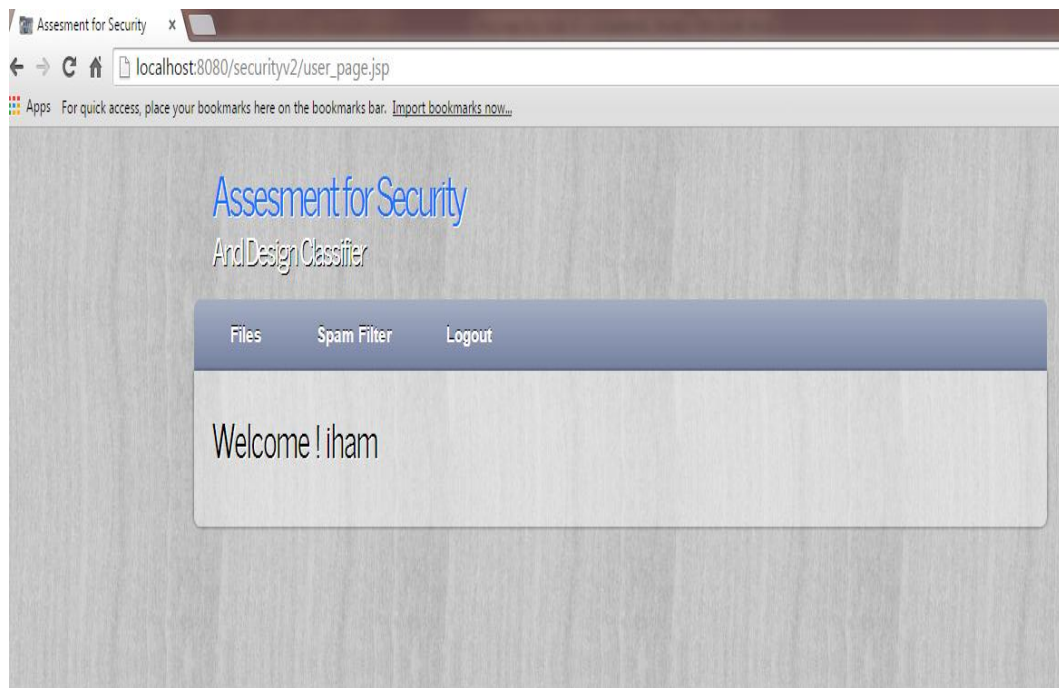
## Email page



(Figure 5.6 Email Page)

From this page user can send or receive Emails.

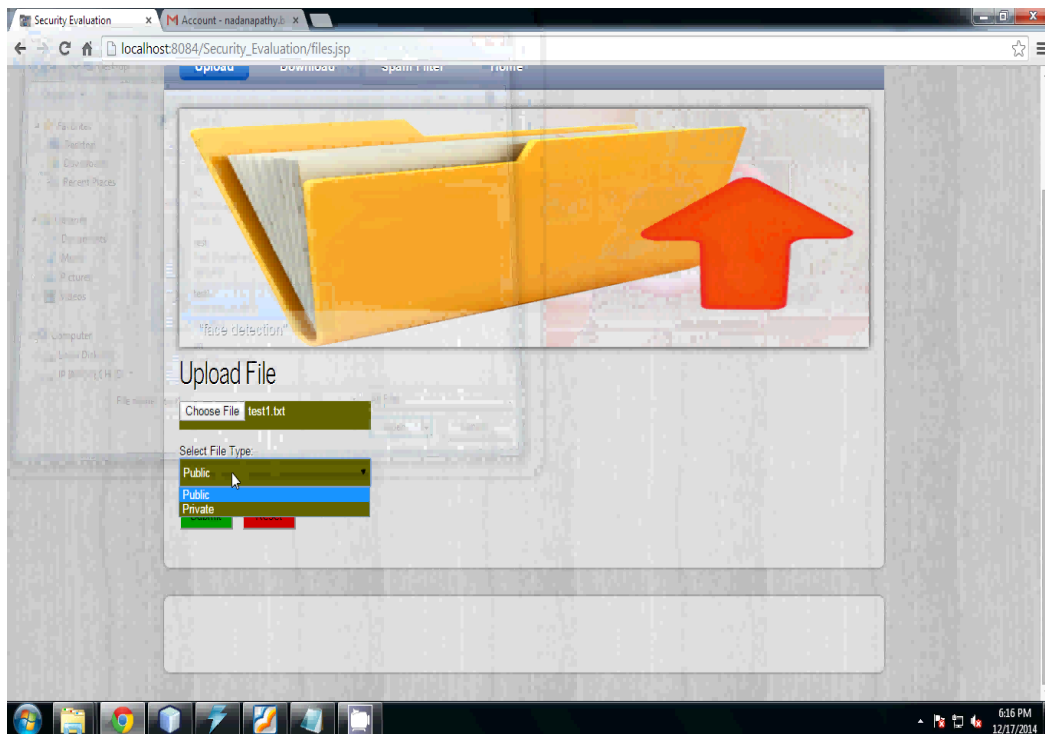
## User Welcome Page



(Figure 5.7 User Welcome Page)

This Page allow user to upload/download the file, apply spam filter on file.

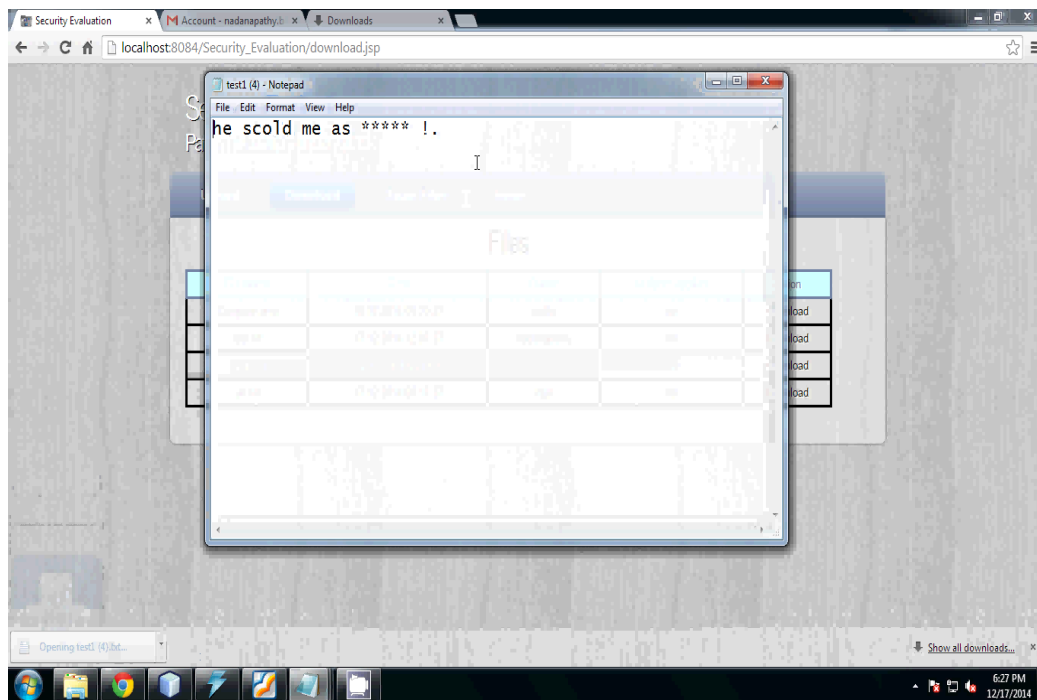
## Public And Private Cloud Choice Screen



(Figure 5.8 Public and Private Cloud File Uploads)

In this page user can choose to which cloud has to upload a specific files either public or private cloud , if the file is in private cloud only user himself can access despite of public cloud file which is available for all the users.

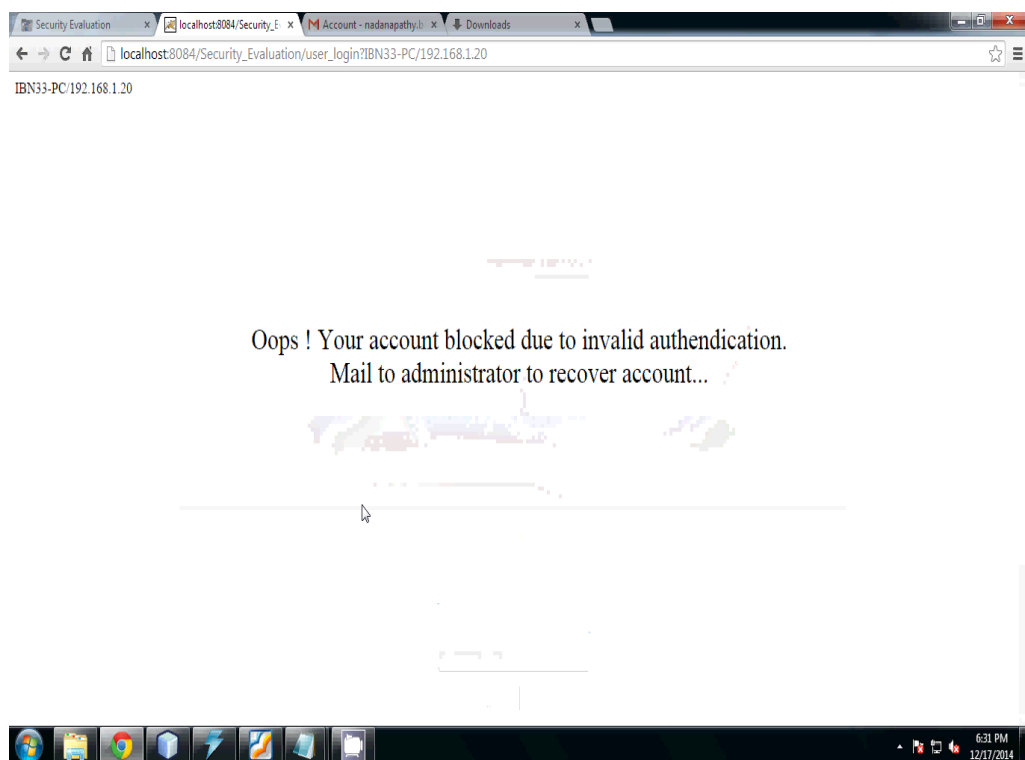
## Display of Spam Filtered File



(Figure 5.9 Display Content of A File)

Display of file after spam filter is applied, the abusing words are replaced by set of stars and the non abusing text are displayed normally.

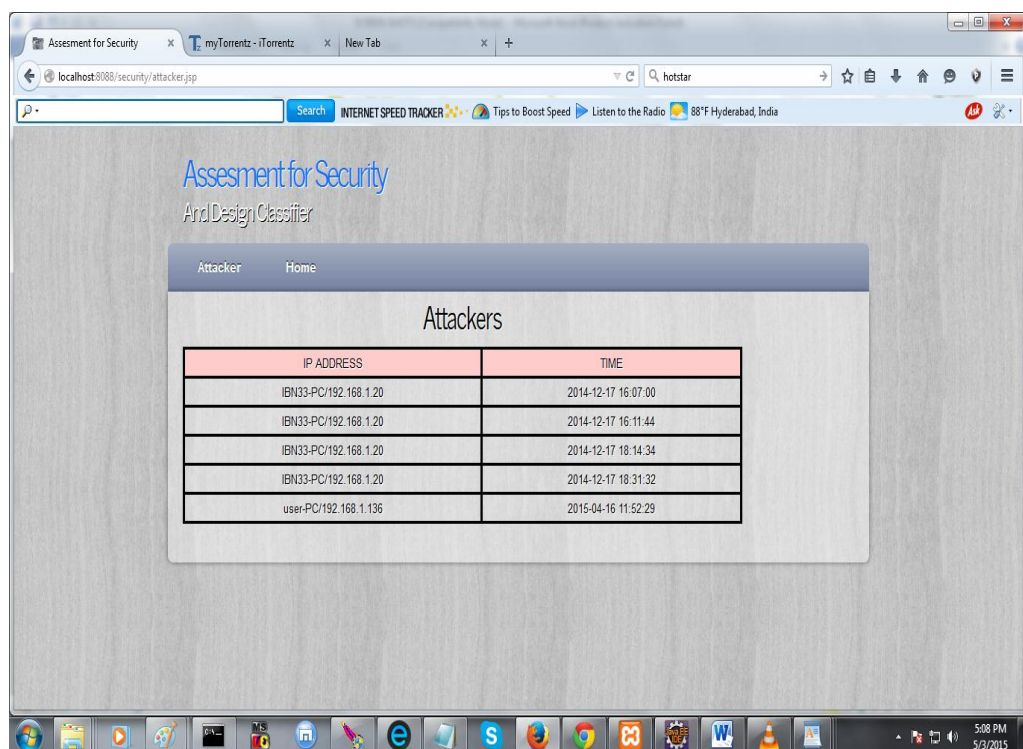
## Warning Page for attackers



(Figure 5.10 Warning for Intruderss)

If a suser is attempting more than three times with wrong credentials , automatically the application blocks his account, after that only admin can grant access again.

## Attackers Details



IP ADDRESS	TIME
IBN33-PC/192.168.1.20	2014-12-17 16:07:00
IBN33-PC/192.168.1.20	2014-12-17 16:11:44
IBN33-PC/192.168.1.20	2014-12-17 18:14:34
IBN33-PC/192.168.1.20	2014-12-17 18:31:32
user-PC/192.168.1.136	2015-04-16 11:52:29

(Figure 5.11 List of suspicious attempters)

Suspicious attempter's system details and time snapshots for attempts are listed here, so admin can easily identify who is trying to break the security.



### 5.3 Sample of source code

#### 5.3.1 Admin Login action

```
<%  
  
try  
  
{  
  
String uname=request.getParameter("username");  
  
String pass=request.getParameter("password");  
  
if(uname.equalsIgnoreCase("admin")&&pass.equalsIgnoreCase("a  
dmin"))  
  
{  
  
//out.println("success... ");  
  
response.sendRedirect("admin_page.jsp");  
  
}  
  
Else  
  
{  
  
out.println("incorrect username or password ");
```

```
}  
  
}  
  
catch(Exception e){  
  
out.println(e);  
  
}  
  
%>
```

### 5.3.2 User Registration Action

```
<% @page import="pack.MailUtil"%>  
  
<% @page import="java.io.FileInputStream"%>  
  
<% @page import="java.io.InputStream"%>  
  
<% @page import="java.io.File"%>  
  
<% @page import="com.oreilly.servlet.MultipartRequest"%>  
  
<% @page import="java.sql.ResultSet"%>  
  
<% @page import="java.sql.PreparedStatement"%>  
  
<% @page import="java.sql.Connection"%>  
  
<% @page import="java.sql.Statement"%>  
  
<% @page import="pack.Dbconnection"%>
```

```
<%  
  
try  
  
{  
  
File file;  
  
final String filepath="D:/";  
  
MultipartRequest m=new MultipartRequest(request,filepath);  
  
file=m.getFile("image");  
  
InputStream is=new FileInputStream(file);  
  
String name=m.getParameter("name");  
  
String uname=m.getParameter("username");  
  
String pass=m.getParameter("password");  
  
String mail=m.getParameter("mail");  
  
String ph=m.getParameter("mobile");  
  
//out.println("name:"+name);  
  
Connection con= Dbconnection.getConn();  
  
Statement st=con.createStatement();  
  
ResultSet rt=st.executeQuery("select * from user_reg where  
  
username='"+uname+"'");
```

```
if(rt.next()){

// out.println("username already exist...");

response.sendRedirect("registration.jsp?exist='username'");

}

Else

{

PreparedStatement ps=con.prepareStatement("insert into user_reg

( username,name,password,mail,phoneno,activate,image_name,

image_count,profile_picture,picture_name)

values(?,?,?,?,?,?,?,?,?,?,?)");

ps.setString(1, uname);

ps.setString(2,name);

ps.setString(3,pass);

ps.setString(4, mail);

ps.setString(5, ph);

ps.setString(6, "no");

ps.setString(7, file.getName());

ps.setAsciiStream(8,is,(int)file.length());
```

```
ps.setString(9, "0");

ps.setAsciiStream(10,is,(int)file.length());

ps.setString(11, file.getName());

boolean s=ps.execute();

String[] mm=new String[]{mail};

new MailUtil().sendMail(mm, mm, "Registration","your account
registered .Use this mailid for any future references...");

if(s){

out.println("internal error...try again later");

}

else{

response.sendRedirect("registration.jsp?yes='registered'");

}

}

}

catch(Exception e){out.println(e);

}

%>
```

### 5.3.3 User Page

```
<% @page import="java.sql.Statement"%>

<% @page import="pack.Dbconnection"%>

<% @page import="java.sql.Connection"%>

<!DOCTYPE html>

<html>

<head>

<title>Security Evaluation</title>

<meta name="description" content="website description" />

<meta name="keywords" content="website keywords, website
keywords" />

<meta http-equiv="content-type" content="text/html;
charset=windows-1252" />

<link rel="stylesheet" type="text/css" href="css/style.css" />

<!-- modernizr enables HTML5 elements and feature detects -->

<script type="text/javascript" src="js/modernizr-
1.5.min.js"></script>
```

```
<link rel="shortcut icon" type="x-icon" href="images/ic.png"/>

<script type="text/javascript">

window.history.forward(1);

function noBack(){

window.history.forward();

}

</script>

</head>

<body onload="noBack();" onpageshow="if (event.persisted)

noBack();" onunload="">

<div id="main">

<header>

<div id="banner">

<div id="welcome">

<h3>Security <span>Evaluation</span></h3>

</div><!--close welcome-->

<div id="welcome_slogan">

<h3>Pattern Classifiers under Attack</h3>
```

```
</div><!--close welcome_slogan-->

</div><!--close banner-->

</header>

<nav>

<div id="menubar">

<ul id="nav">

<li><a href="files.jsp">Files</a></li>

<li><a href="spam_filter.jsp">Spam Filter</a></li>

<li><a href="profile_picture.jsp">Change Profile Picture</a></li>

<li><a href="index.html">Logout</a></li>

<!--      <li><a href="projects.html">Projects</a></li>

<li><a href="contact.html">Contact Us</a></li>-->

</ul>

</div><!--close menubar-->

</nav>

<div id="site_content">

<%

Connection con= Dbconnection.getConn();
```



```
Statement st=con.createStatement();

%>

<div >

<ul >

<li ></li>

<!--      <li></li>-->

<!--      <li></li>-->

</ul>

</div>

<div id="content">

<div class="content_item">

<%

HttpSession user=request.getSession();

String name=user.getAttribute("name").toString();
```

```
%>

<h1>Welcome ! <%=name%></h1>

</div><!--close content_item-->

</div><!--close content-->

</div><!--close site_content-->

</div><!--close main-->

<!-- <footer>

<a href="index.html">Home</a> | <a href="ourwork.html">Our
Work</a> | <a href="testimonials.html">Testimonials</a> | <a
href="projects.html">Projects</a> | <a
href="contact.html">Contact</a><br/><br/>

website template by <a
href="http://www.freehtml5templates.co.uk">Free HTML5
Templates</a>

</footer>-->

<!-- javascript at the bottom for fast page loading -->

<script type="text/javascript" src="js/jquery.js"></script>
```

```
<script type="text/javascript" src="js/image_slide.js"></script>  
  
</body>  
  
</html>
```

### 5.3.4 Spam Filter

```
<% @page import="java.sql.ResultSet"%>  
  
<% @page import="java.sql.Statement"%>  
  
<% @page import="pack.Dbconnection"%>  
  
<% @page import="java.sql.Connection"%>  
  
<!DOCTYPE html>  
  
<html>  
  
<head>  
  
<title>Security Evaluation</title>  
  
<meta name="description" content="website description" />  
  
<meta name="keywords" content="website keywords, website  
keywords" />  
  
<meta http-equiv="content-type" content="text/html;
```

```
charset=windows-1252" />

<link rel="stylesheet" type="text/css" href="css/style.css" />

<!-- modernizr enables HTML5 elements and feature detects -->

<script type="text/javascript" src="js/modernizr-

1.5.min.js"></script>

<link rel="shortcut icon" type="x-icon" href="images/ic.png"/>

<!--      <script type="text/javascript">

window.history.forward(1);

function noBack(){

    window.history.forward();

}

</script>-->

<style>

#id{

width: 200px;

height: 25px;

background-color: #D3F2F7;

}

#but{
```

```
width: 70px;

height: 25px;

}

table{

width: 900px;

}

td{

text-align: center;

}

table,td,tr{

border-style: solid;

border-collapse: collapse;

}

tr{

height: 30px;

}

</style>

<script>
```

```
function val(){  
  
    if(confirm("Are sure you wanna apply spam filter to this file...")){  
  
    }  
  
    else{  
  
        return false;  
  
    }  
  
    }  
  
</script>  
  
</head>  
  
<!--<body onload="noBack();" onpageshow="if (event.persisted)  
noBack();" onunload="">-->  
  
<body>  
  
<%  
  
if(request.getParameter("yes")!=null){  
  
    out.println("<script>alert('spam filter applied...')</script>");  
  
    }  
  
}
```

```
%>

<div id="main">

<header>

<div id="banner">

<div id="welcome">

<h3>Security <span>Evaluation</span></h3>

</div><!--close welcome-->

<div id="welcome_slogan">

<h3>Pattern Classifiers under Attack</h3>

</div><!--close welcome_slogan-->

</div><!--close banner-->

</header>

<nav>

<div id="menubar">

<ul id="nav">

<li><a href="files.jsp">Upload</a></li>

<li ><a href="download.jsp">Download</a></li>

<li class="current"><a href="spam_filter.jsp">Spam
```

```
Filter</a></li>

<li><a href="user_page.jsp">Home</a></li>

<!--      <li><a href="projects.html">Projects</a></li>

<li><a href="contact.html">Contact Us</a></li>-->

</ul>

</div><!--close menubar-->

</nav>

<div id="site_content">

<!--      <div class="slideshow">

<ul class="slideshow">

<li class="show"></li>

<li></li>

<li></li>

</ul>

</div>      -->
```



```
<div id="content">

<div class="content_item">

<%

HttpSession user=request.getSession();

String name=user.getAttribute("name").toString();

String uname=user.getAttribute("username").toString();

Connection con=Dbconnection.getConn();

Statement st=con.createStatement();

ResultSet rt=st.executeQuery("select * from files where

owner_='"+uname+"'");

%>

<!--      <h1>Welcome ! <%=name%></h1> -->

<table>

<caption> <h1> Files</h1> </caption>

<tr style="background-color: #ccffff;color: #7380A0">

<td>File Name</td><td>Time</td><td>Owner</td><td>Is Spam

applied</td><td>Action</td>

</tr>
```

```
<%  
  
while(rt.next()){  
  
String id=rt.getString("id");  
  
%>  
  
<tr>  
  
<td><%=rt.getString("file_name")%></td>  
  
<td><%=rt.getString("time_")%></td>  
  
<td><%=rt.getString("owner_")%></td>  
  
<td><%=rt.getString("applied_spam")%></td>  
  
<td><a href="spam.jsp?<%=id%>" onclick="return val();">Apply  
Spam Filter</a></td>  
  
<!--          <td><a href="spam_filter?12" onclick="return  
val()">Apply Spam Filter</a></td>-->  
  
</tr>  
  
<%  
  
}  
  
%>  
  
</table>
```

```
</div><!--close content_item-->

</div><!--close content-->

</div><!--close site_content-->

</div><!--close main-->

<!-- <footer>

<a href="index.html">Home</a> | <a href="ourwork.html">Our
Work</a> | <a href="testimonials.html">Testimonials</a> | <a
href="projects.html">Projects</a> | <a
href="contact.html">Contact</a><br/><br/>
website template by <a
href="http://www.freehtml5templates.co.uk">Free HTML5
Templates</a> </footer>-->

<!-- javascript at the bottom for fast page loading -->

<script type="text/javascript" src="js/jquery.js"></script>

<script type="text/javascript" src="js/image_slide.js"></script>

</body>

</html>
```

**Chapter Six**  
**System Testing**

## **6 System Testing**

### **6.1 Testing**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

### **6.2 Types of Test**

#### **6.2.1 Unit Test**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the

application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

### **6.2.2 Integration Test**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

### **6.2.3 Functional Test**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked. Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

#### **6.2.4 System Test**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

### **6.2.5 White Box Testing**

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

### **6.2.6 Black Box Testing**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.



## **Unit Testing**

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

### **Test strategy and approach**

Field testing will be performed manually and functional tests will be written in detail.

### **Test objectives**

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

### **Features to be tested**

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

**Integration Testing:**

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

**Test Results**

All the test cases mentioned above passed successfully. No defects encountered.

**Acceptance Testing**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**Test Results**

All the test cases mentioned above passed successfully. No defects encountered.

**Chapter Seven  
Conclusion**

## Result

In this paper we focused on empirical security evaluation of pattern classifiers that have to be deployed in adversarial environments, and proposed how to revise the classical performance evaluation design step, which is not suitable for this purpose.

Our main contribution is a framework for empirical security evaluation that formalizes and generalizes ideas from previous work, and can be applied to different classifiers, learning algorithms, and classification tasks. It is grounded on a formal model of the adversary, and on a model of data distribution that can represent all the attacks considered in previous work; provides a systematic method for the generation of training and testing sets that enables security evaluation; and can accommodate application-specific techniques for attack simulation. This is a clear advancement with respect to previous work, since without a general framework most of the proposed techniques (often tailored to a given classifier model, attack, and application) could not be directly applied to other problems.

An intrinsic limitation of our work is that security evaluation is carried out empirically, and it is thus datadependent; on the other hand, model-driven analyses [12], [17], [38] require a full analytical model of the problem and of the adversary's behavior, that may be very difficult to develop for real-world applications. Another intrinsic limitation is due to fact that our method is not application-specific, and, therefore, provides only high-level guidelines for simulating attacks. Indeed, detailed

guidelines require one to take into account application-specific constraints and adversary models. Our future work will be devoted to develop techniques for simulating attacks for different applications.

Although the design of secure classifiers is a distinct problem than security evaluation, our framework could be also exploited to this end. For instance, simulated attack samples can be included into the training data to improve security of discriminative classifiers (e.g., SVMs), while the proposed data model can be exploited to design more secure generative classifiers. We obtained encouraging preliminary results on this topic

**Indexes****A**

Activity Diagram, 6, 65  
attack, 2, 4, 12, 13, 14, 20, 57,  
67, 69, 99, 100  
Authentication, 108, 109

**B**

Black Box, 6, 95

**C**

class, 2, 13, 31, 35, 41, 44, 63,  
69, 86, 88, 89  
Class Diagram, 6, 63  
classification, 4, 2, 7, 19, 21, 67,  
68, 99

**D**

Data Mining, 5, 6, 8, 9, 104, 105

**E**

Existing System, 5, 19

**F**

Feasibility, 5, 15, 16, 17  
Filter, 85, 86, 88, 89, 106

**I**

Intrusion, 69, 105, 106, 108, 109

**J**

J2EE, 24  
JAVA, 24

**M**

MYSQL, 24

**N**

Network, 41, 109

**P**

Pattern, 1, 2, 4, 2, 19, 67, 68, 85,  
88, 103, 107, 109, 110  
Proposed System, 5, 20, 21

**S**

Sequence Diagram, 6, 64  
Spam, 13, 14, 85, 86, 88, 89, 103,  
106, 108, 110  
system analysis, 16  
System Design, 6, 56  
System Requirements, 5, 24

**U**

use case diagram, 61

**W**

White Box, 6, 94

## **Bibliography**

- [1] R.N. Rodrigues, L.L. Ling, and V. Govindaraju, "Robustness of Multimodal Biometric Fusion Methods against Spoof Attacks," *J. Visual Languages and Computing*, vol. 20, no. 3, pp. 169-179, 2009.
- [2] P. Johnson, B. Tan, and S. Schuckers, "Multimodal Fusion Vulnerability to Non-Zero Effort (Spoof) Imposters," *Proc. IEEE Int'l Workshop Information Forensics and Security*, pp. 1-5, 2010.
- [3] P. Fogla, M. Sharif, R. Perdisci, O. Kolesnikov, and W. Lee, "Polymorphic Blending Attacks," *Proc. 15th Conf. USENIX Security Symp.*, 2006.
- [4] G.L. Wittel and S.F. Wu, "On Attacking Statistical Spam Filters," *Proc. First Conf. Email and Anti-Spam*, 2004.
- [5] D. Lowd and C. Meek, "Good Word Attacks on Statistical Spam Filters," *Proc. Second Conf. Email and Anti-Spam*, 2005.
- [6] A. Kolcz and C.H. Teo, "Feature Weighting for Improved Classifier Robustness," *Proc. Sixth Conf. Email and Anti-Spam*, 2009.
- [7] D.B. Skillicorn, "Adversarial Knowledge Discovery," *IEEE Intelligent Systems*, vol. 24, no. 6, Nov./Dec. 2009.

- [8] D. Fetterly, “Adversarial Information Retrieval: The Manipulation of Web Content,” *ACM Computing Rev.*, 2007.
- [9] R.O. Duda, P.E. Hart, and D.G. Stork, *Pattern Classification*. Wiley-Interscience Publication, 2000.
- [10] N. Dalvi, P. Domingos, Mausam, S. Sanghai, and D. Verma, “Adversarial Classification,” *Proc. 10th ACM SIGKDD Int’l Conf. Knowledge Discovery and Data Mining*, pp. 99-108, 2004.
- [11] M. Barreno, B. Nelson, R. Sears, A.D. Joseph, and J.D. Tygar, “Can Machine Learning be Secure?” *Proc. ACM Symp. Information, Computer and Comm. Security (ASIACCS)*, pp. 16-25, 2006.
- [12] A.A. Cardenas and J.S. Baras, “Evaluation of Classifiers: Practical Considerations for Security Applications,” *Proc. AAAI Workshop Evaluation Methods for Machine Learning*, 2006.
- [13] P. Laskov and R. Lippmann, “Machine Learning in Adversarial Environments,” *Machine Learning*, vol. 81, pp. 115-119, 2010.
- [14] L. Huang, A.D. Joseph, B. Nelson, B. Rubinstein, and J.D. Tygar, “Adversarial Machine Learning,” *Proc. Fourth ACM Workshop Artificial Intelligence and Security*, pp. 43-57, 2011.
- [15] M. Barreno, B. Nelson, A. Joseph, and J. Tygar, “The Security of Machine Learning,” *Machine Learning*, vol. 81, pp. 121-148, 2010.



- [16] D. Lowd and C. Meek, "Adversarial Learning," Proc. 11th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining, pp. 641-647, 2005.
- [17] P. Laskov and M. Kloft, "A Framework for Quantitative Security Analysis of Machine Learning," Proc. Second ACM Workshop Security and Artificial Intelligence, pp. 1-4, 2009.
- [18] NIPS Workshop Machine Learning in Adversarial Environments for Computer Security, <http://mls-nips07.first.fraunhofer.de/>, 2007.
- [19] Dagstuhl Perspectives Workshop Mach. Learning Methods for Computer Sec., <http://www.dagstuhl.de/12371/>, 2012.
- [20] A.M. Narasimhamurthy and L.I. Kuncheva, "A Framework for Generating Data to Simulate Changing Environments," Proc. 25<sup>th</sup> Conf. Proc. the 25th IASTED Int'l Multi-Conf.: Artificial Intelligence and Applications, pp. 415-420, 2007.
- [21] S. Rizzi, "What-If Analysis," Encyclopedia of Database Systems, pp. 3525-3529, Springer, 2009.
- [22] J. Newsome, B. Karp, and D. Song, "Paragraph: Thwarting Signature Learning by Training Maliciously," Proc. Ninth Int'l Conf. Recent Advances in Intrusion Detection, pp. 81-105, 2006.

- [23] A. Globerson and S.T. Roweis, “Nightmare at Test Time: Robust Learning by Feature Deletion,” Proc. 23rd Int’l Conf. Machine Learning, pp. 353-360, 2006.
- [24] R. Perdisci, G. Gu, and W. Lee, “Using an Ensemble of One-Class SVM Classifiers to Harden Payload-Based Anomaly Detection Systems,” Proc. Int’l Conf. Data Mining, pp. 488-498, 2006.
- [25] S.P. Chung and A.K. Mok, “Advanced Allergy attacks: Does a Corpus Really Help,” Proc. 10th Int’l Conf. Recent Advances in Intrusion Detection (RAID ’07), pp. 236-255, 2007.
- [26] Z. Jorgensen, Y. Zhou, and M. Inge, “A Multiple Instance Learning Strategy for Combating Good Word Attacks on Spam Filters,” J. Machine Learning Research, vol. 9, pp. 1115-1146, 2008.
- [27] G.F. Cretu, A. Stavrou, M.E. Locasto, S.J. Stolfo, and A.D. Keromytis, “Casting out Demons: Sanitizing Training Data for Anomaly Sensors,” Proc. IEEE Symp. Security and Privacy, pp. 81-95, 2008.
- [28] B. Nelson, M. Barreno, F.J. Chi, A.D. Joseph, B.I.P. Rubinstein, U. Saini, C. Sutton, J.D. Tygar, and K. Xia, “Exploiting Machine Learning to Subvert Your Spam Filter,” Proc. First Workshop Large- Scale Exploits and Emergent Threats, pp. 1-9, 2008.
- [29] B.I. Rubinstein, B. Nelson, L. Huang, A.D. Joseph, S.-h. Lau, S. Rao, N. Taft, and J.D. Tygar, “Antidote: Understanding and Defending

against Poisoning of Anomaly Detectors,” Proc. Ninth ACM SIGCOMM Internet Measurement Conf. (IMC '09), pp. 1-14, 2009.

[30] M. Kloft and P. Laskov, “Online Anomaly Detection under Adversarial Impact,” Proc. 13th Int’l Conf. Artificial Intelligence and Statistics, pp. 405-412, 2010.

[31] O. Dekel, O. Shamir, and L. Xiao, “Learning to Classify with Missing and Corrupted Features,” Machine Learning, vol. 81, pp. 149-178, 2010.

[32] B. Biggio, G. Fumera, and F. Roli, “Design of Robust Classifiers for Adversarial Environments,” Proc. IEEE Int’l Conf. Systems, Man, and Cybernetics, pp. 977-982, 2011.

[33] B. Biggio, G. Fumera, and F. Roli, “Multiple Classifier Systems for Robust Classifier Design in Adversarial Environments,” Int’l J. Machine Learning and Cybernetics, vol. 1, no. 1, pp. 27-41, 2010.

[34] B. Biggio, I. Corona, G. Fumera, G. Giacinto, and F. Roli, “Bagging Classifiers for Fighting Poisoning Attacks in Adversarial Environments,” Proc. 10th Int’l Workshop Multiple Classifier Systems, pp. 350-359, 2011.

[35] B. Biggio, G. Fumera, F. Roli, and L. Didaci, “Poisoning Adaptive Biometric Systems,” Proc. Joint IAPR Int’l Conf. Structural, Syntactic, and Statistical Pattern Recognition, pp. 417-425, 2012.

[36] B. Biggio, B. Nelson, and P. Laskov, "Poisoning Attacks against Support Vector Machines," Proc. 29th Int'l Conf. Machine Learning, 2012.

[37] M. Kearns and M. Li, "Learning in the Presence of Malicious Errors," SIAM J. Computing, vol. 22, no. 4, pp. 807-837, 1993.

[38] A.A. Cardenas, J.S. Baras, and K. Seamon, "A Framework for the Evaluation of Intrusion Detection Systems," Proc. IEEE Symp. Security and Privacy, pp. 63-77, 2006.

[39] B. Biggio, G. Fumera, and F. Roli, "Multiple Classifier Systems for Adversarial Classification Tasks," Proc. Eighth Int'l Workshop Multiple Classifier Systems, pp. 132-141, 2009.

[40] M. Brückner, C. Kanzow, and T. Scheffer, "Static Prediction Games for Adversarial Learning Problems," J. Machine Learning Research, vol. 13, pp. 2617-2654, 2012.

[41] A. Adler, "Vulnerabilities in Biometric Encryption Systems," Proc. Fifth Int'l Conf. Audio- and Video-Based Biometric Person Authentication, pp. 1100-1109, 2005.

[42] B. Efron and R.J. Tibshirani, An Introduction to the Bootstrap. Chapman & Hall, 1993.

- [43] H. Drucker, D. Wu, and V.N. Vapnik, "Support Vector Machines for Spam Categorization," IEEE Trans. Neural Networks, vol. 10, no. 5, pp. 1048-1054, Sept. 1999.
- [44] F. Sebastiani, "Machine Learning in Automated Text Categorization," ACM Computing Surveys, vol. 34, pp. 1-47, 2002.
- [45] C.-C. Chang, C.-J. Lin, "LibSVM: A Library for Support Vector Machines," <http://www.csie.ntu.edu.tw/~cjlin/libsvm/>, 2001.
- [46] K. Nandakumar, Y. Chen, S.C. Dass, and A. Jain, "Likelihood Ratio-Based Biometric Score Fusion," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 30, no. 2, pp. 342-347, Feb. 2008.
- [47] B. Biggio, Z. Akhtar, G. Fumera, G. Marcialis, and F. Roli, "Robustness of Multi-Modal Biometric Verification Systems under Realistic Spoofing Attacks," Proc. Int'l Joint Conf. Biometrics, pp. 1-6, 2011.
- [48] B. Biggio, Z. Akhtar, G. Fumera, G.L. Marcialis, and F. Roli, "Security Evaluation of Biometric Authentication Systems under Real Spoofing Attacks," IET Biometrics, vol. 1, no. 1, pp. 11-24, 2012.
- [49] K. Wang and S.J. Stolfo, "Anomalous Payload-Based Network Intrusion Detection," Proc. Seventh Symp. Recent Advances in Intrusion Detection (RAID), pp. 203-222, 2004.

- [50] B. Schölkopf, A.J. Smola, R.C. Williamson, and P.L. Bartlett, "New Support Vector Algorithms," *Neural Computation*, vol. 12, no. 5, pp. 1207-1245, 2000.
- [51] K. Ingham and H. Inoue, "Comparing Anomaly Detection Techniques for http," *Proc. 10th Int'l Conf. Recent Advances in Intrusion Detection*, pp. 42-62, 2007.
- [52] D. Sculley, G. Wachman, and C.E. Brodley, "Spam Filtering Using Inexact String Matching in Explicit Feature Space with on-Line Linear Classifiers," *Proc. 15th Text Retrieval Conf.*, 2006.
- [53] *Encyclopedia of Biometrics*, S.Z. Li, and A.K. Jain, eds., Springer US, 2009.
- [54] B. Biggio, G. Fumera, and F. Roli, "Adversarial Pattern Classification Using Multiple Classifiers and Randomisation," *Proc. Joint IAPR Int'l Workshop Structural, Syntactic, and Statistical Pattern Recognition*, pp. 500-509, 2008.