

COMPROMISED ACCOUNT DETECTION ON SOCIAL NETWORKS

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering Degree in Computer Science and Engineering

By

ALLADI PHANI KRISHNA

(38110023)

GURAJALA HARI HARA VAMSI KUMAR

(38110184)



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited with Grade "A" by NAAC

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600119

MAY-2022



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC

(Established under Section 3 of UGC Act, 1956)

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI- 600119

www.sathyabamauniversity.ac.in



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **ALLADI PHANI KRISHNA(38110023), GURAJALA HARI HARA VAMSI KUMAR(38110184)** who carried out the project entitled "**COMPROMISED ACCOUNT DETECTION ON SOCIAL NETWORKS**" under my supervision from November 2021 to April 2022.

Internal Guide

Dr. P. SARDAR MARAN M.E., Ph.D.

Head of the Department

Dr .S. VIGNESHWARI, M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I ALLADI PHANI KRISHNA(38110023),GURAJALA HARI HARA VAMSI KUMAR(38110184) hereby declare that the Project Report entitled COMPROMISED ACCOUNT DETECTION ON SOCIAL NETWORKS Done by me under the guidance of Dr. P. Sardar Maran M.E., Ph.D at is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE:

SIGNATURE OF THECANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph.D., Dean**, School of Computing , **Dr.S.Vigneshwari M.E., Ph.D., and Dr.L.Lakshmanan M.E., Ph.D.,** Heads of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide

Dr.P. Sardar Maran M.E.,Ph.D for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Hacking virtual entertainment accounts has turned into a helpful method for completing crimes. Programmers can take well known media or business records to get out awful word or gotten out bogus word to numerous clients. The outcome of the calamity cost the monetary business sectors many billions of dollars. In our past work, we told the best way to get the confusions of informal community clients. In these works, we have told the best way to decide the consistency of a specific record utilizing comparable innovation. Famous records have one trademark that makes this idea solid - they show a steady person over the long haul. That's what we show assuming our framework is executed, it is feasible to identify and forestall three genuine assaults on trustworthy organizations and media. Also, not at all like other famous news sources, our framework doesn't fall into the class of misconceptions brought about by the American chain of eateries.

Keywords: Behavioral analysis, cyber security, Text mining, social network analysis

TABLE OF CONTENTS

CHAPTER No.	TITLE	PAGE No
	ABSTRACT	5
1	INTRODUCTION	7
2.	LITERATURE SURVEY	8
3.	AIM AND SCOPE OF PRESENT	8
	3.1 AIM	8
	3.2 SCOPE	8
	3.3 WORK FLOW DIAGRAM	9
4.	SOFTWARE ENVIRONMENT	14
	4.1 APACHE	14
	4.2 PHP	15
	4.3 MySQL	16
	4.4 AJAX	
5	MODULES	22
	5.1 BEHAVIOURAL PROFILES	22
	5.2 MODELLING CHARACTERISTICS	23
	5.3 TRAINING EVALUTION OF THE MODELS	23
	5.4 BEHAVIOURAL PROFILE STABILITY	23
6	SCREENSCHOTS & DIAGRAMS	24
7	SOURCE CODE AND REFERENCE	38

INTRODUCTION

Online social networks, such as Facebook and Twitter, have become one of the main media to stay in touch with the rest of the world. Celebrities use them to communicate with their fan base, corporations take advantage of them to promote their brands and have a direct connection to their customers, while news agencies leverage social networks to distribute breaking news. Regular users make pervasive use of social networks too, to stay in touch with their friends or colleagues and share content that they find interesting. Over time, social network users build trust relationships with the accounts they follow. This trust can develop for a variety of reasons. For example, the user might know the owner of the trusted account in person or the account might be operated by an entity commonly considered as trustworthy, such as a popular news agency. Unfortunately, should the control over an account fall into the hands of a cyber-criminal, he can easily exploit this trust to further his own malicious agenda. Previous research showed that using compromised accounts to spread malicious content is advantageous to cyber criminals, because social network users are more likely to react to messages coming from accounts they trust. These favorable probabilities of success exceedingly attract the attention of cyber criminals. Once an attacker compromises a social network account he can use it for nefarious purposes such as sending spam messages or link to malware and phishing web sites. Such traditional attacks are best carried out through a large population of compromised accounts belonging to regular social network account users. Recent incidents, however, demonstrate that attackers can cause havoc and interference even by compromising individual, but highprofile accounts. These accounts (e.g., newspaper or popular brand name accounts) have large social circles (i.e., followers) and their popularity suggests trustworthiness to many social network users. Recent attacks show that compromising these high profile accounts can be leveraged to disseminate fake news alerts, or messages that tarnish a company's reputation.

CHAPTER 2

LITERATURE SURVEY

Literature survey is the most important step in software development process. Before developing the tool it is necessary to determine the time factor, economy and company strength. Once these things are satisfied, then the next step is to determine which operating system and language can be used for developing the tool. Once the programmers start building the tool the programmers need lot of external support. This support can be obtained from senior programmers, from book or from websites. Before building the system the above consideration are taken into account for developing the proposed system. The major part of the project development sector considers and fully survey all the required needs for developing the project. For every project Literature survey is the most important sector in software development process. Before developing the tools and the associated designing it is necessary to determine and survey the time factor, resource requirement, man power, economy, and company strength. Once these things are satisfied and fully surveyed, then the next step is to determine about the software specifications in the respective system such as what type of operating system the project would require, and what are all the necessary software are needed to proceed with the next step such as developing the tools, and the associated operations.

CHAPTER 3

AIM AND SCOPE OF PRESENT INVESTIGATION

3.1 AIM:

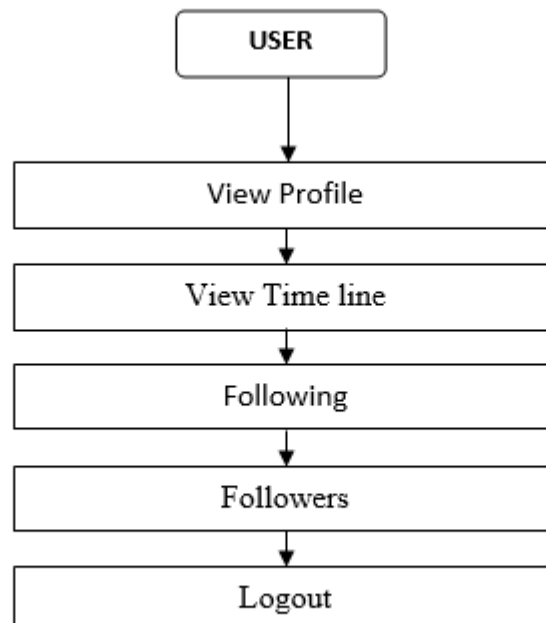
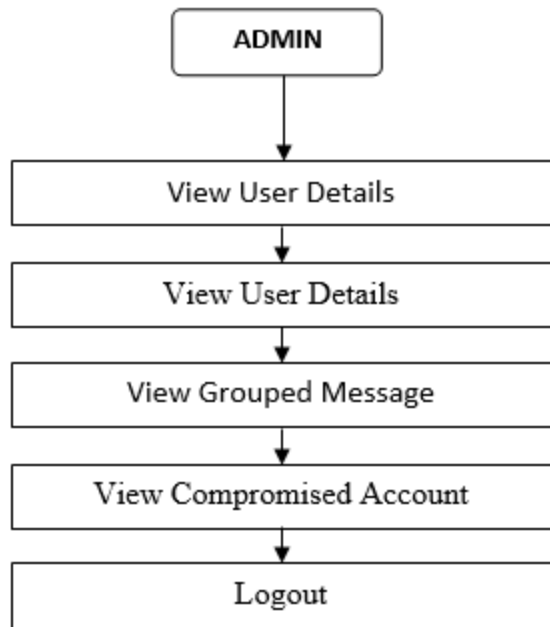
To detect the compromised accounts on social networks

3.2 SCOPE:

* End to End compromised account detection

* Semantic text analysis

3.3 WORK FLOW DIAGRAM



End-to-End Compromised account detection

Social media, e.g. Twitter, has become a widely used medium for the exchange of information, but it has also become a valuable tool for hackers to spread misinformation through compromised accounts. Hence, detecting compromised accounts is a necessary step toward a safe and secure social media environment. Nevertheless, detecting compromised accounts faces several challenges. First, social media activities of users are temporally correlated which plays an important role in compromised account detection. Second, data associated with social media accounts is inherently sparse. Finally, social contagions where multiple accounts become compromised, take advantage of the user connectivity to propagate their attack. Thus how to represent each user's network features for compromised account detection is an additional challenge. To address these challenges, we propose an End-to-End Compromised Account Detection framework (E2ECAD). E2ECAD effectively captures temporal correlations via an LSTM (Long Short-Term Memory) network. Further, it addresses the sparsity problem by defining and employing a user context representation. Meanwhile, informative network-related features are modeled efficiently. To verify the working of the framework, we construct a real-world dataset of compromised accounts on Twitter and conduct extensive experiments. The results of experiments show that E2ECAD outperforms the state of the art compromised account detection algorithms.

Semantic Text Analysis for Detection of Compromised Accounts on Social Networks

Compromised accounts on social networks are regular user accounts that have been taken over by an entity with malicious intent. Since the adversary exploits the already established trust of a compromised account, it is crucial to detect these accounts to limit the damage they can cause. We propose a novel general framework for semantic analysis of text messages coming out from an account to detect compromised accounts. Our framework is built on the observation that normal users will use language that is measurably different from the language that an adversary would use when the account is compromised. We propose to use the difference of language models of users and adversaries to define novel interpretable semantic features for measuring semantic

incoherence in a message stream. We study the effectiveness of the proposed semantic features using a Twitter data set. Evaluation results show that the proposed framework is effective for discovering compromised accounts on social networks and a KL-divergence-based language model feature works best.

EXISTING SYSTEM

Thomas et al. built Monarch to detect malicious messages on social networks based on URLs that link to malicious sites. By relying only on URLs, Monarch misses other types of malicious messages. For example, our previous work illustrates that COMPA detects scams based on phone numbers and XSS worms spreading without linking to a malicious URL. WARNINGBIRD is a system that detects spam links posted on Twitter by analyzing the characteristics of HTTP redirection chains that lead to a final spam page. Xu et al. present a system that, by monitoring a small number of nodes, detects worms propagating on social networks. This paper does not directly address the problem of compromised accounts, but could detect large-scale infections such as koobface. Yang et al. studied new Twitter spammers that act in a stealthy way to avoid detection. In their system, they use advanced features such as the topology of the network that surrounds the spammer. They do not try to distinguish compromised from spam accounts.

DISADVANTAGES OF EXISTING SYSTEM

- Previous systems will fail to recognize them as malicious
- This approach does not scale as it requires identifying and joining each new phishing campaign. Also, this approach is limited to phishing campaigns
- Previous work showed that spam victims occasionally send messages to these

spam accounts. This would cause their approach to detect legitimate accounts as compromised.

- Is not as generic.

PROPOSED SYSTEM

In this paper we present COMPA, the first detection system designed to identify compromised social network accounts. COMPA is based on a simple observation: social network users develop habits over time, and these habits are fairly stable. A typical social network user, for example, might consistently check her posts in the morning from her phone, and during the lunch break from her desktop computer. Furthermore, interaction will likely be limited to a moderate number of social network contacts (i.e., friends). Conversely, if the account falls under the control of an adversary, the messages that the attacker sends will likely show anomalies compared to the typical behavior of the user. To detect account compromises, COMPA builds a behavioral profile for social network accounts, based on the messages sent by the account in the past. Every time a new message is generated, the message is compared against this behavioral profile. If the message significantly deviates from the learned behavioral profile, COMPA flags it as a possible compromise. In this paper we first show that high profile accounts often have well-defined behavioral profiles that allow COMPA to detect compromises with very low false positives. However, behavioral profiles of regular user accounts are more variable than their well-defined counterparts of most high profile accounts. This is because regular users are more likely to experiment with new features or client software to engage with the social network. This variability could cause an increase of false positive alerts. However, social network accounts of regular users are less influential than high profile accounts. Thus, attackers aggregate multiple accounts into a campaign to achieve effects that are similar to the compromise of a high profile account.

ADVANTAGES OF PROPOSED SYSTEM

- We present COMPA, the first system designed to detect compromised social network accounts.
- We show that COMPA can reliably detect compromises that affect high profile accounts. Since the behavior of these accounts is very consistent, false positives are minimal.
- To detect large-scale compromises, we propose to group similar messages together and apply COMPA to them, to assess how many of those messages violate their accounts' behavioral profile. This grouping accounts for the fact that regular social network accounts show a more variable behavior compared to high profile ones, and allows us to keep false positives low.

SYSTEM REQUIREMENTS

HARDWARE REQUIREMENTS

- System : Pentium Dual Core.
- Hard Disk : 120 GB.
- Monitor : 15" LED
- Input Devices : Keyboard, Mouse
- Ram : 1 GB

SOFTWARE REQUIREMENTS

- Operating system : Windows 7.
- Coding Language : PHP
- Database : MYSQL

CHAPTER 4 SOFTWARE ENVIRONMENT

Language Description

Web Server Introduction

Web servers are computers on the Internet that host websites, serving pages to viewers upon request. This service is referred to as web hosting. Every web server has a unique address so that other computers connected to the internet know where to find it on the vast network. The Internet Protocol (IP) address looks something like this: 65.65.120.35. This address maps to a more human friendly address, such as <http://www.xyz.com>. Web servers stay connected to the Internet 24 hours a day, seven days a week, 365 days a year. In truth, they experience occasional downtime due to maintenance and technical problems. Web servers with consistent records of an uptime of 99.5% or better are considered reliable.

4.1:Apache

The Apache HTTP Server commonly referred to as Apache, is web server

software notable for playing a key role in the initial growth of the World Wide Web. In 2009 it became the first web server software to surpass the 100 million web site milestone. Apache supports a variety of features, many implemented as compiled modules which extend the core functionality. These can range from server-side programming language support to authentication schemes. Some common language interfaces support Perl, Python, Tcl, and PHP. Virtual hosting allows one Apache installation to serve many different actual websites. For example, one machine with one Apache installation could simultaneously serve www.xyz1.com, www.xyz2.com, and etc.

Apache features configurable error messages, DBMS-based authentication databases, and content negotiation. It is also supported by several graphical user interfaces (GUIs).

4.2:PHP

Hypertext Preprocessor is a widely used, general-purpose scripting language that was originally designed for web development to produce dynamic web pages. For this purpose, PHP code is embedded into the HTML source document and interpreted by a web server with a PHP processor module, which generates the web page document. PHP is available as a processor for most modern web servers and as standalone interpreter on most operating systems and computing platforms. It is a free, open-source server-side scripting language. PHP code can be embedded in HTML. PHP files usually have extensions like `.php` or `.php3`. PHP language style is similar to C and Java.

4.3:MySQL

MySQL is a relational database management system (RDBMS) that runs as a server providing multi-user access to a number of databases. Free-software projects that require a full-featured database management system often use MySQL. Such projects include (for example) Word Press, phpBB, Drupal and other software built on the LAMP software stack. MySQL is also used in many high-profiles, large-scale World Wide Web products including Wikipedia, Google and Face book. We listed a few features of it as below:

- Stored Procedures, Triggers, Views
- Query Caching
- Full Text Indexing
- Partial Unicode Support
- SSL Support
- Embedded database library
- Replication support

4.4:AJAX

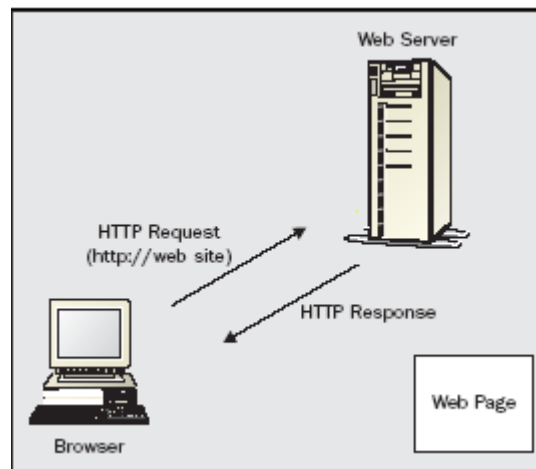
AJAX (Asynchronous JavaScript and XML) is a group of interrelated web development techniques used on the client-side to create interactive web applications or

rich Internet applications. With Ajax, web applications can retrieve data from the server asynchronously in the background without interfering with the display and behavior of the existing page. The use of Ajax has led to an increase in interactive animation on web pages and better quality of Web services.

Technology Specification

Client Server Architecture

The server (the web server) stores, interprets, and distributes data, and the client (browser) accesses the server to get at the data. From now on whenever we use the term client, we are just referring to the browser.



Client and server communication without PHP

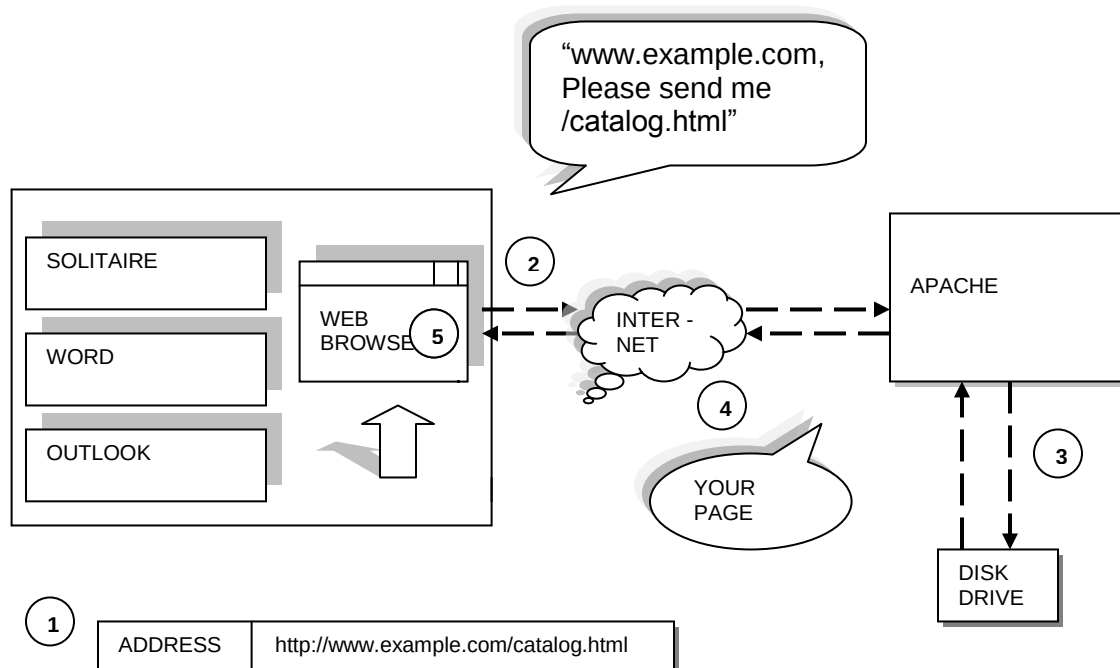
Here's what's happening in the numbered steps of the diagram:

1. If we type `www.example.com/catalog.html` into the location bar of Internet Explorer,
2. Internet Explorer sends a message over the Internet to the computer named `www.example.com` asking for the `/catalog.html` page.

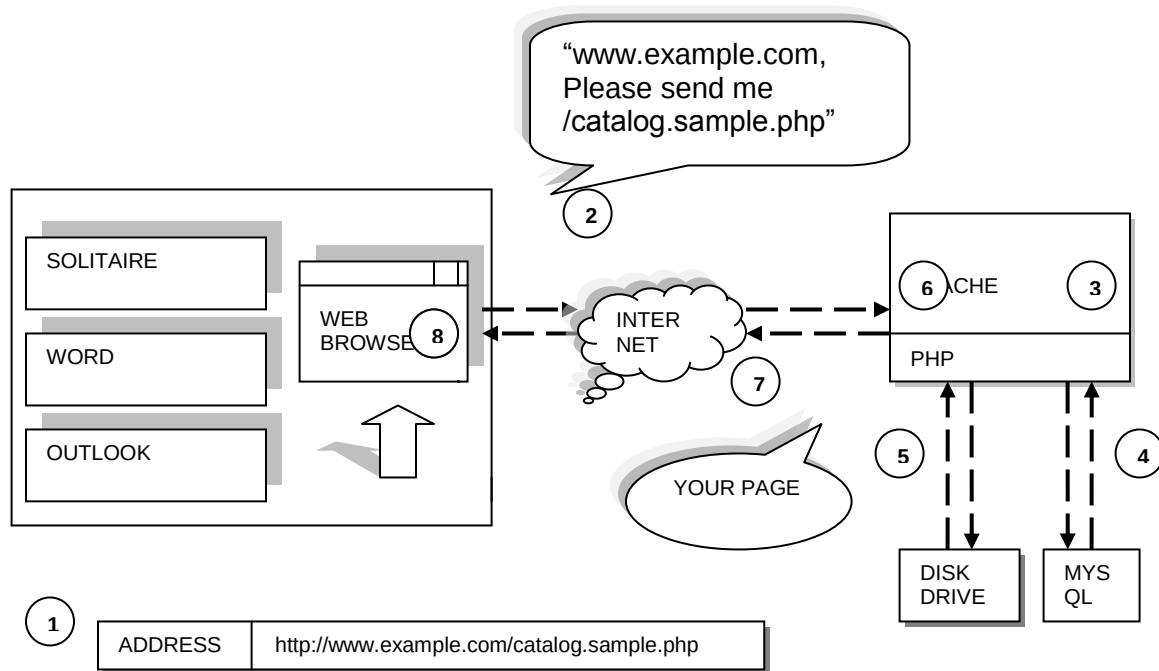
3. Apache, a program running on the www.example.com computer, gets the message and reads the catalog.html file from the disk drive.
4. Apache sends the contents of the file back to our computer over the Internet as a response to Internet Explorer's request.
5. Internet Explorer displays the page on the screen, following the instructions of the HTML tags in the page.

Every time a browser asks for `http://www.example.com/catalog.html`, the web server sends back the contents of the same `catalog.html` file. The only time the response from the web server changes is if someone edits the file on the server.

The following figure shows Client and server communication without PHP



Client and server communication with PHP



1. If we type `www.example.com/catalog/sample.php` into the location bar of Internet Explorer,
2. Internet Explorer sends a message over the Internet to the computer named `www.example.com` asking for the `/catalog/sample.php` page.
3. Apache, a program running on the `www.example.com` computer, gets the message and asks the PHP interpreter, another program running on the `www.example.com` computer, "What does `/catalog/sample.php` look like?"
4. The PHP interpreter reads the file `/usr/local/www/catalog/sample.php` from the disk drive.
5. The PHP interpreter runs the commands in `sample.php`, possibly exchanging data with a database program such as MySQL.

6. The PHP interpreter takes the yak.php program output and sends it back to Apache as an answer to "What does /catalog/ sample.php look like?"
7. Apache sends the page contents it got from the PHP interpreter back to our computer over the Internet in response to Internet Explorer's request.
8. Internet Explorer displays the page on the screen, following the instructions of the HTML tags in the page

CHAPTER 5

MODULES

- Behavioral Profiles
- Modelling Message Characteristics
- Training and Evaluation of the Models
- Behavioral Profile Stability

MODULES DESCRIPTION

5.1: Behavioral Profiles

In this module, a behavioral profile leverages historical information about the activities of a social network user to capture this user's normal (expected) behavior. To build behavioral profiles, our system focuses on the stream of messages that a user has posted on the social network. Of course, other features such as profile pictures or social activity (e.g., establishing friend or follower relationships) could be useful as well. Unfortunately, social networks typically do not offer a way to retrieve historical data about changes in these features, and therefore, we were unable to use them.

A behavioral profile for a user U is built in the following way: Initially, our system obtains the stream of messages of U from the social networking site. The message stream is a list of all messages that the user has posted on the social network, in chronological order. For different social networks, the message streams are collected in slightly different ways. For example, on Twitter, the message stream corresponds to a user's public timeline. For Facebook, the message stream contains the posts a user wrote on her own wall, but it also includes the messages that this user has posted on her friends' walls.

To be able to build a comprehensive profile, the stream needs to contain a minimum amount of messages. Intuitively, a good behavioral profile has to capture the breadth and variety of ways in which a person uses her social network account (e.g., different

client applications or languages).

5.2:Modelling Message Characteristics

Our approach models the following seven features when building a behavioral profile.

Time (hour of day):

This model captures the hour(s) of the day during which an account is typically active. Many users have certain periods during the course of a day where they are more likely to post (e.g., lunch breaks) and others that are typically quiet (e.g., regular sleeping hours). If a user's stream indicates regularities in social network usage, messages that appear during hours that are associated with quiet periods are considered anomalous.

Message Text (Language):

A user is free to author her messages in any language. However, we would expect that each user only writes messages in a few languages (typically, one or two). Thus, especially for profiles where this feature is relatively stable, a change in the language is an indication of a suspicious change in user activity.

To determine the language that a message was written in, we leverage the libtextcat library. This library performs n-gram-based text categorization, as proposed by Cavnar and Trenkle. Of course, for very short messages, it is often difficult to determine the language. This is particularly problematic for Twitter messages, which are limited to at most 140 characters and frequently contain abbreviated words or uncommon spelling.

Message Topic:

Users post many messages that contain chatter or mundane information. But we would also expect that many users have a set of topics that they frequently talk about, such as favorite sports teams, music bands, or TV shows. When users typically focus on a few topics in their messages and then suddenly post about some different and unrelated subject, this new message should be rated as anomalous.

Links in Messages:

Often, messages posted on social networking sites contain links to additional resources, such as blogs, pictures, videos, or news articles. Links in messages of social networks are so common that some previous work has strongly focused on the analysis of URLs, often as the sole factor, to determine whether a message is malicious or not. We also make use of links as part of the behavioral profile of a user.

5.3:Training and Evaluation of the Models

In a nutshell, COMPA works as follows: for each social network user, we retrieve the past messages that the user has authored. We then extract features for each message, and build behavioral models for each feature separately. Then, we assess whether each individual feature is anomalous or not, based on previous observations. Finally, we combine the anomaly scores for each feature to obtain a global anomaly score for each message.

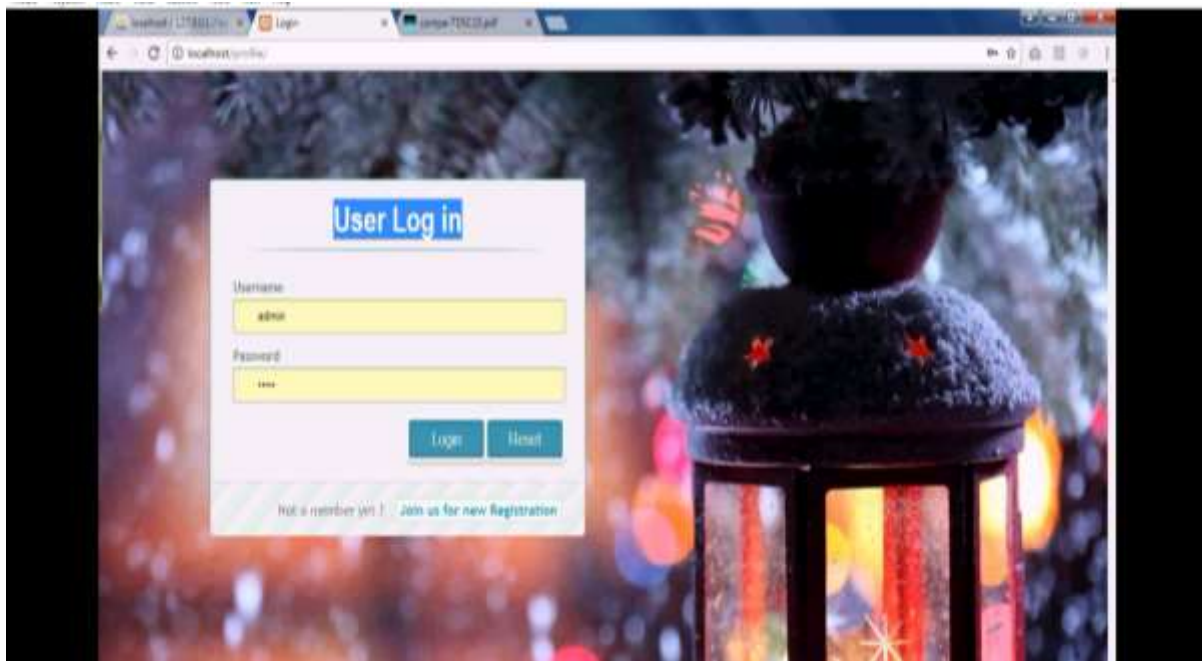
5.4:Behavioral Profile Stability

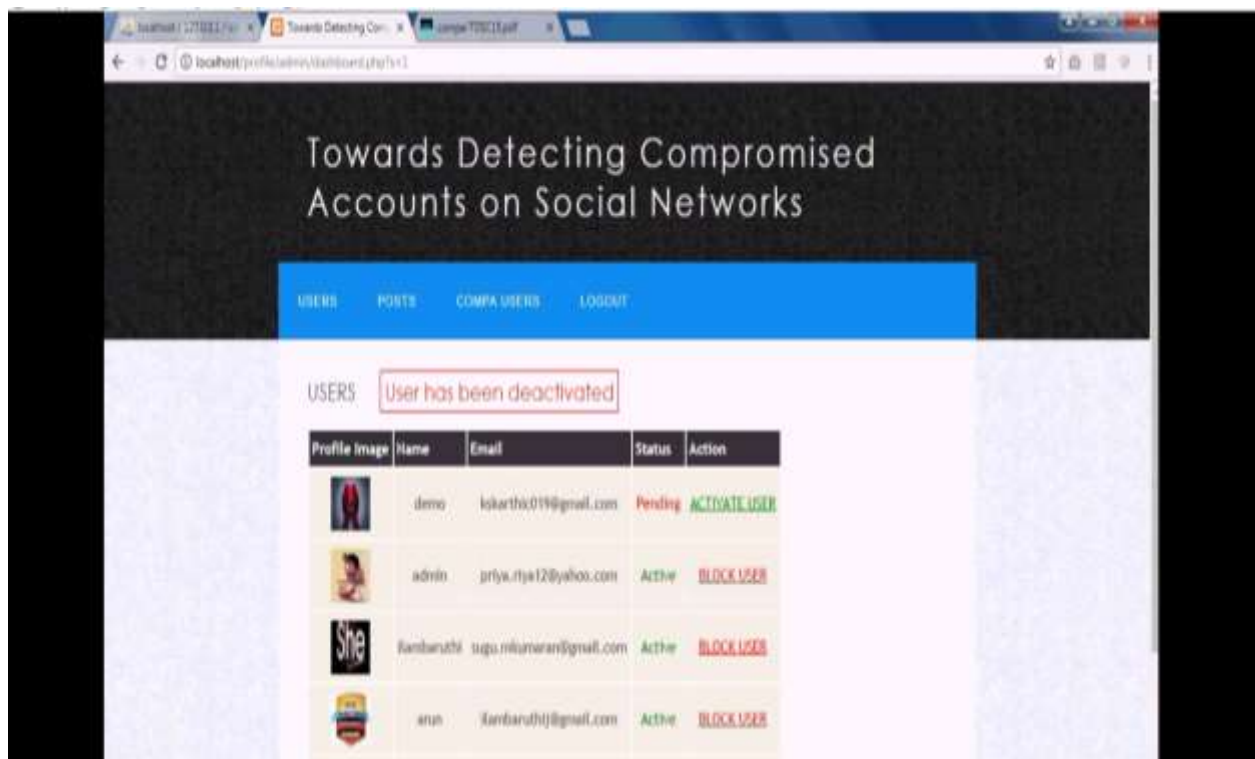
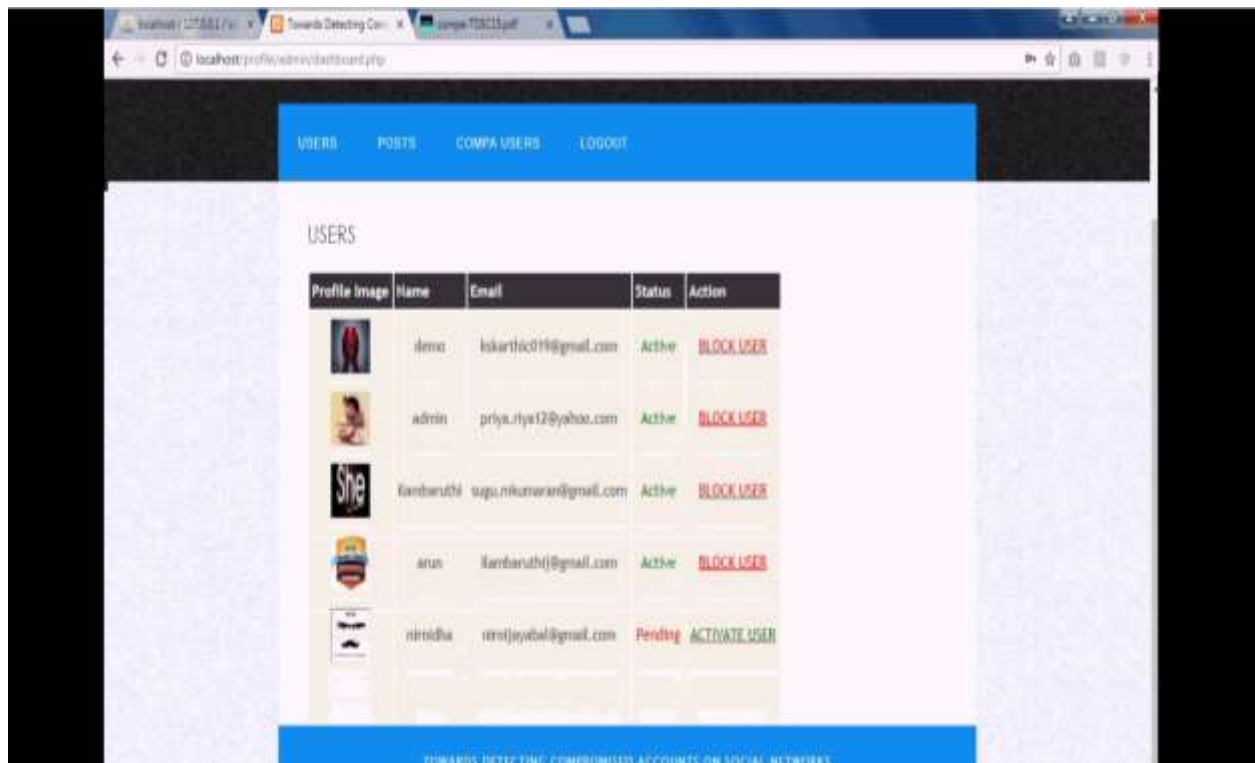
In this module, detecting deviations in account behavior is simplified if the commonly occurring behavior follows mostly regular patterns. Thus, in this section we ask (and

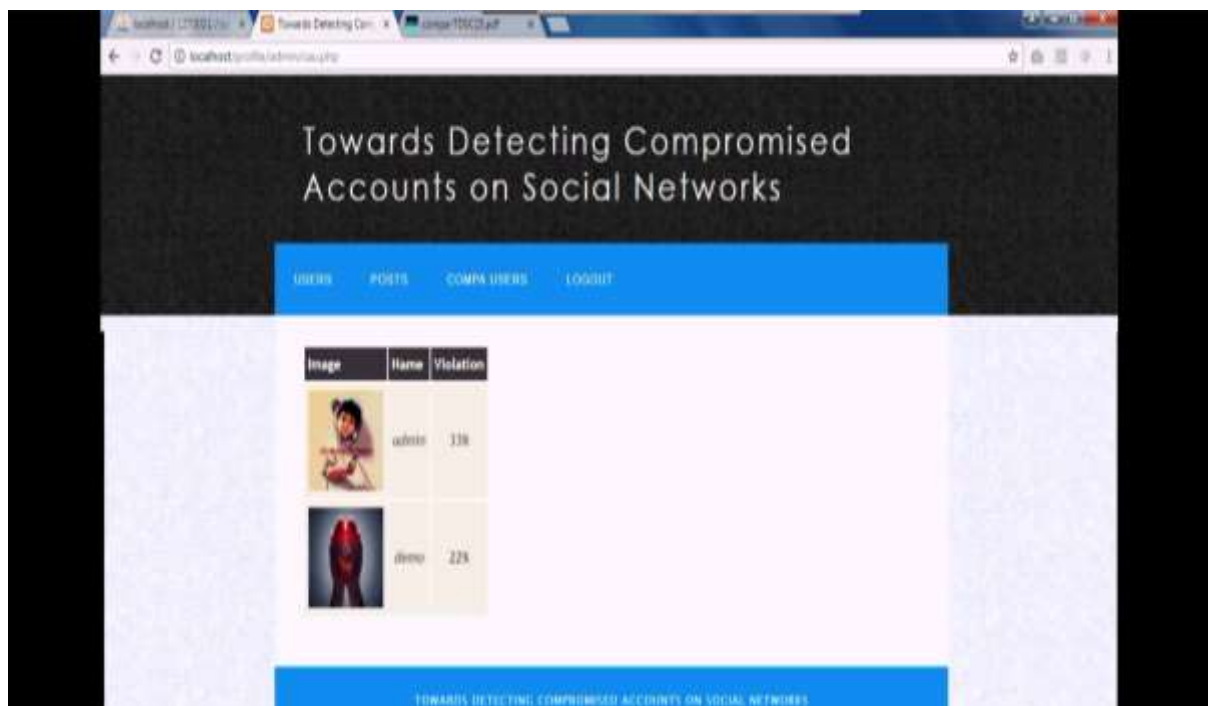
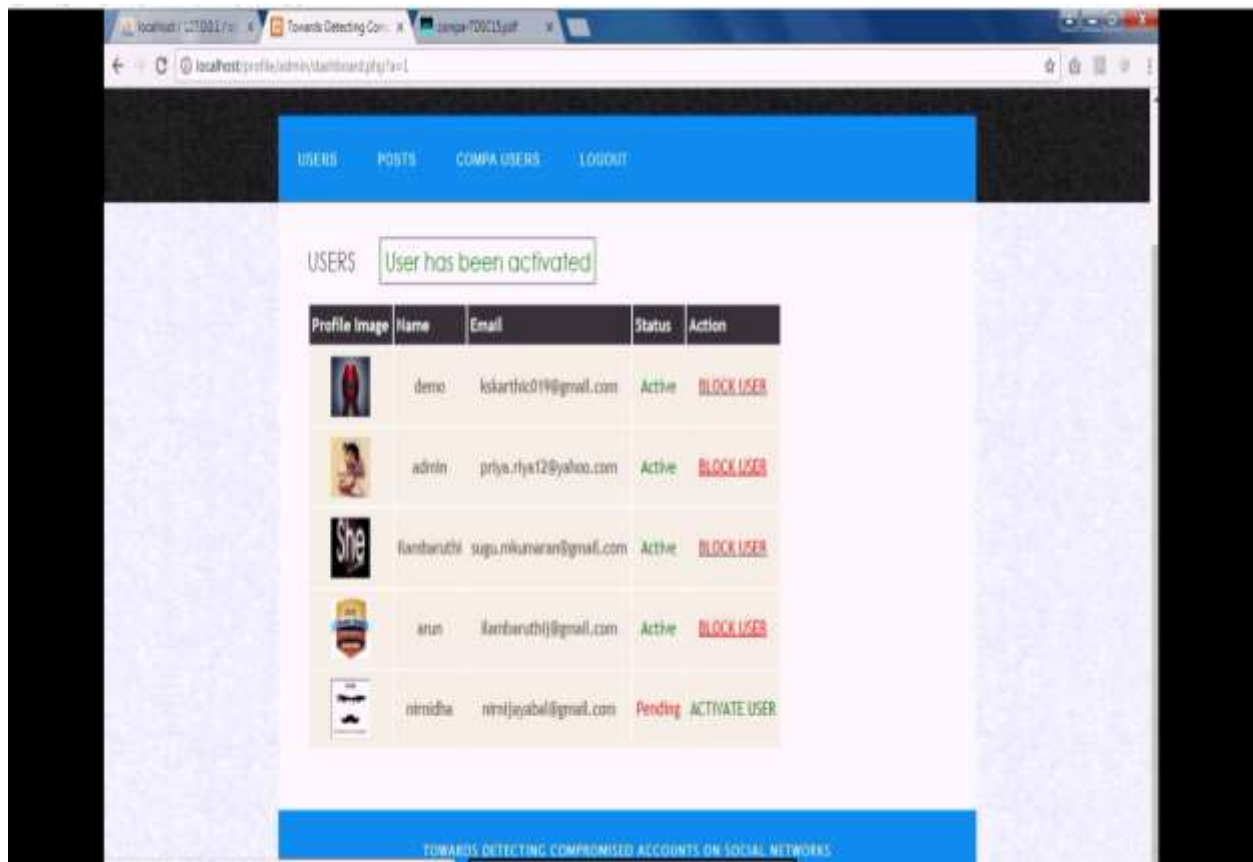
answer) the question of whether there is a class of social network accounts that are particularly amenable to such an analysis. Arguably, a social network strategy is a crucial part for the public relation department of most contemporary companies. Intuitively, we would expect a well-managed company account to show a more stable behavior over time than accounts operated by regular users.

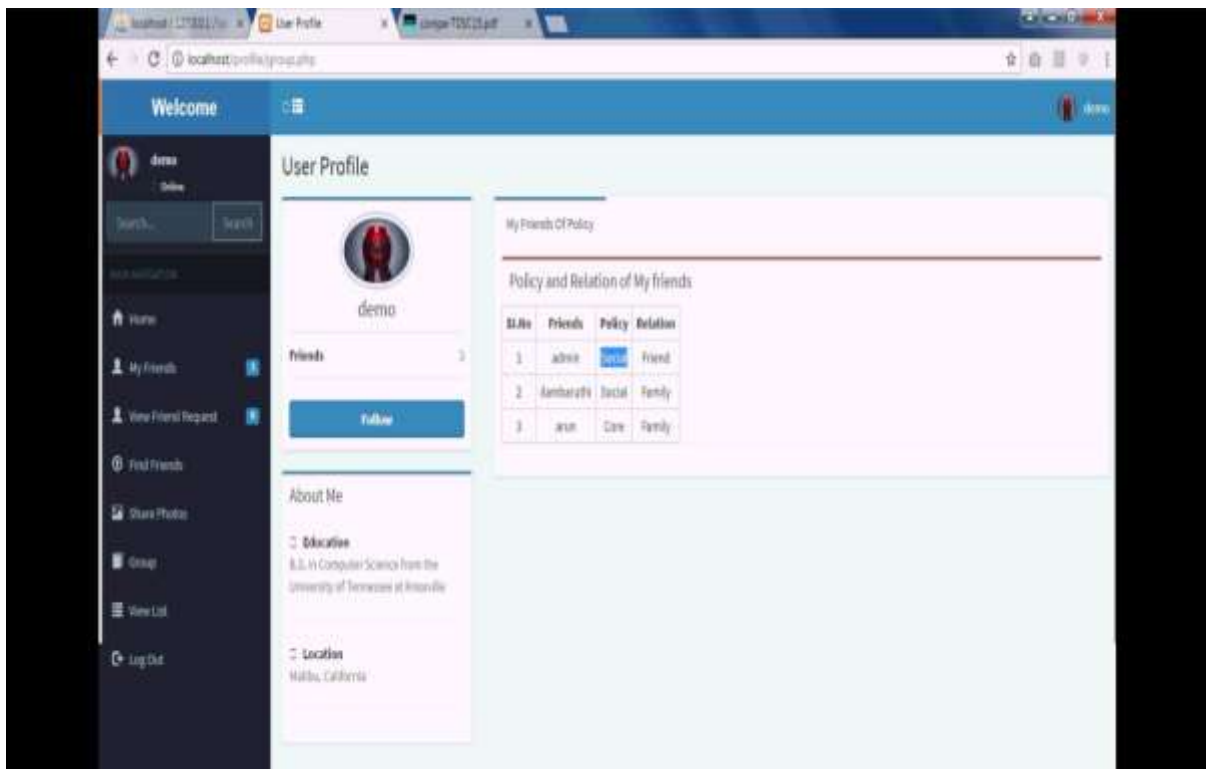
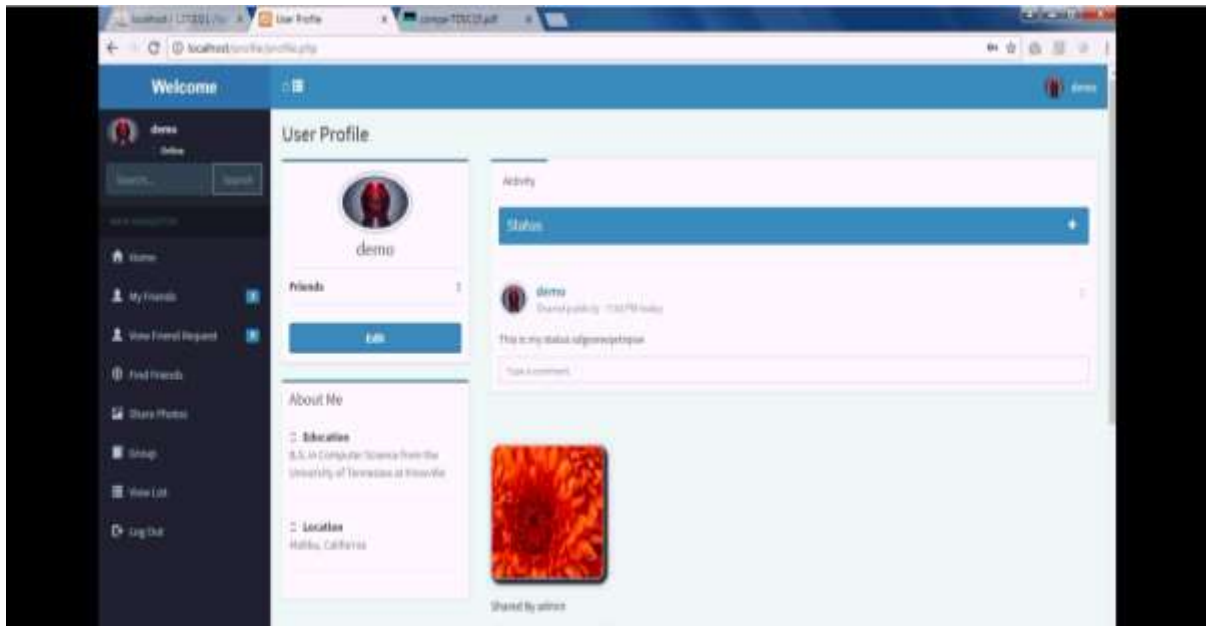
CHAPTER 6

SCREENSHOTS & DIAGRAMS









UML DIAGRAMS

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.

LS:

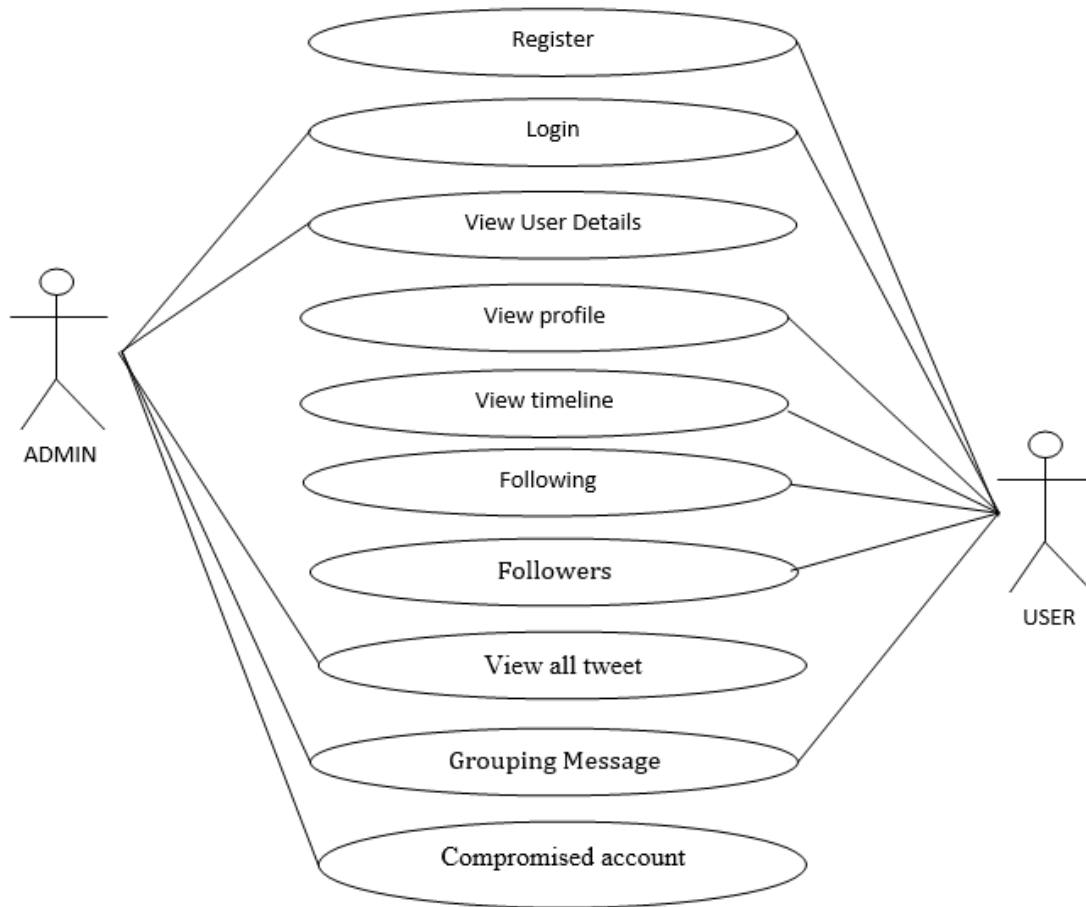
The Primary goals in the design of the UML are as follows:

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.

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

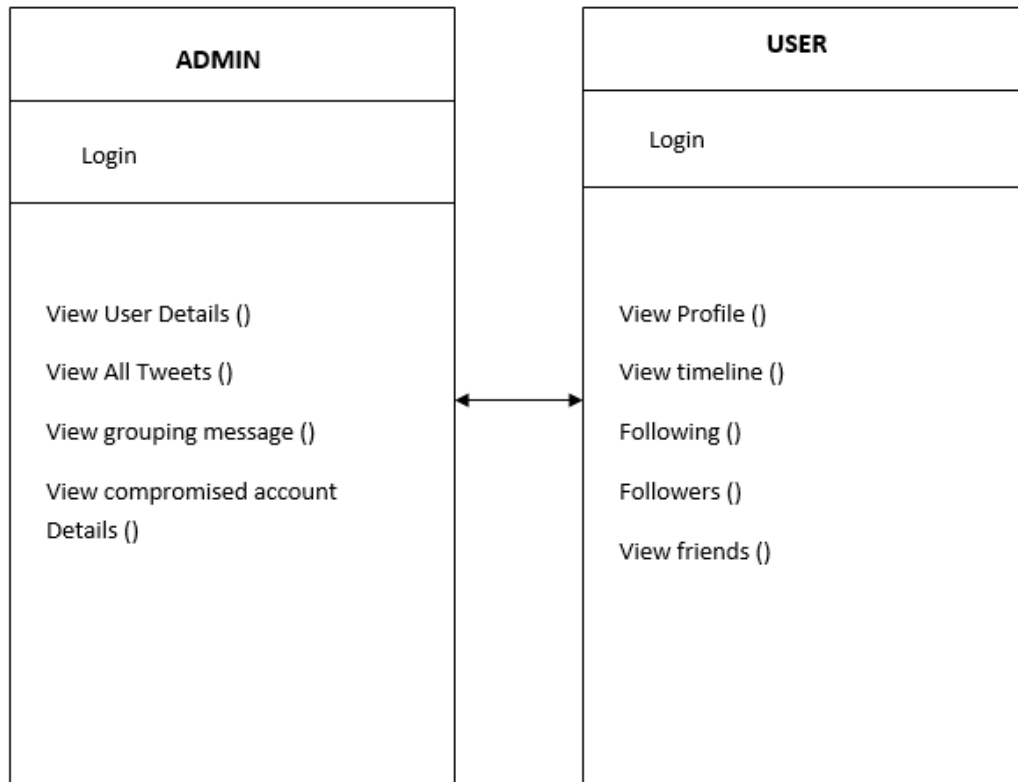
use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



CLASS DIAGRAM:

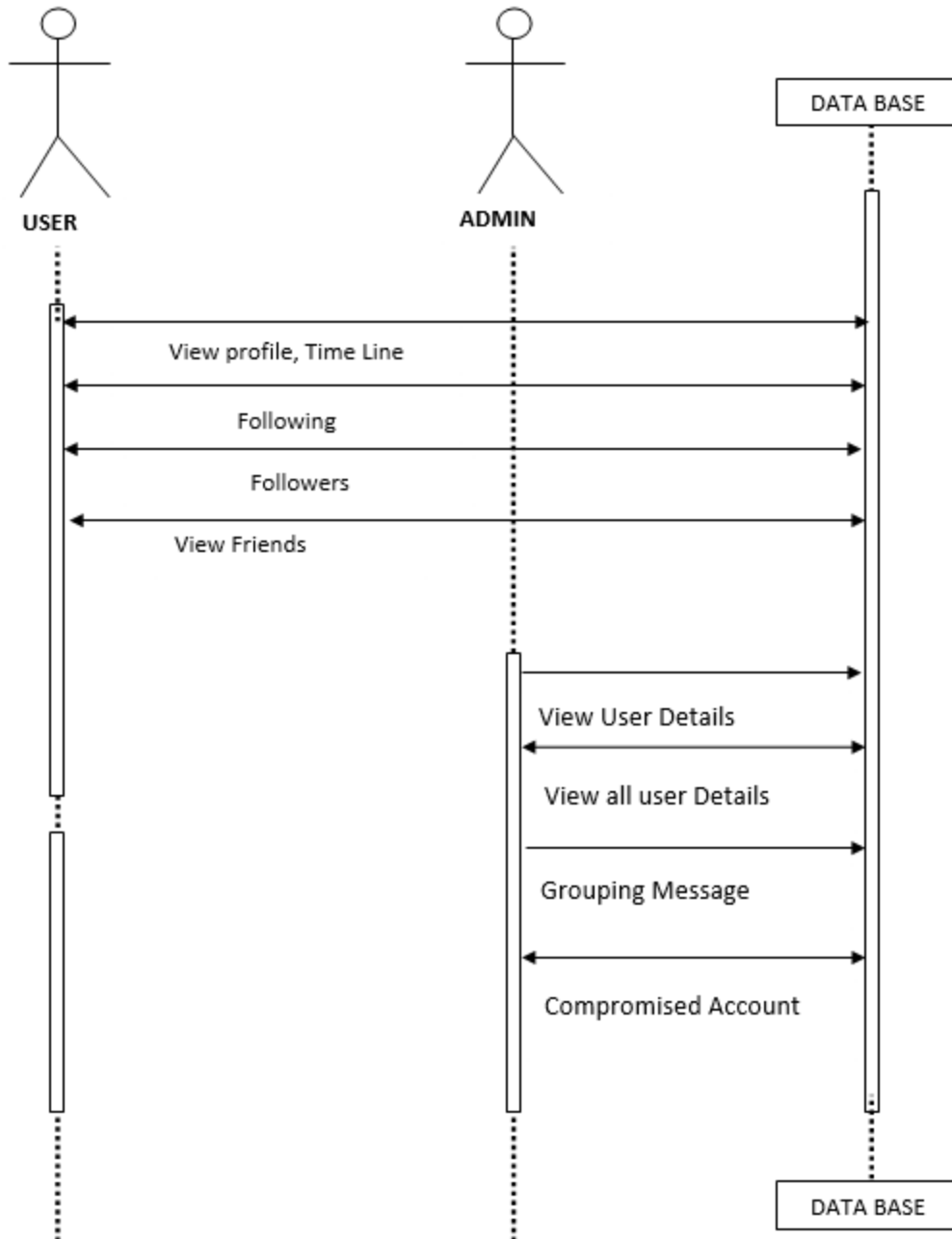
In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the

system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



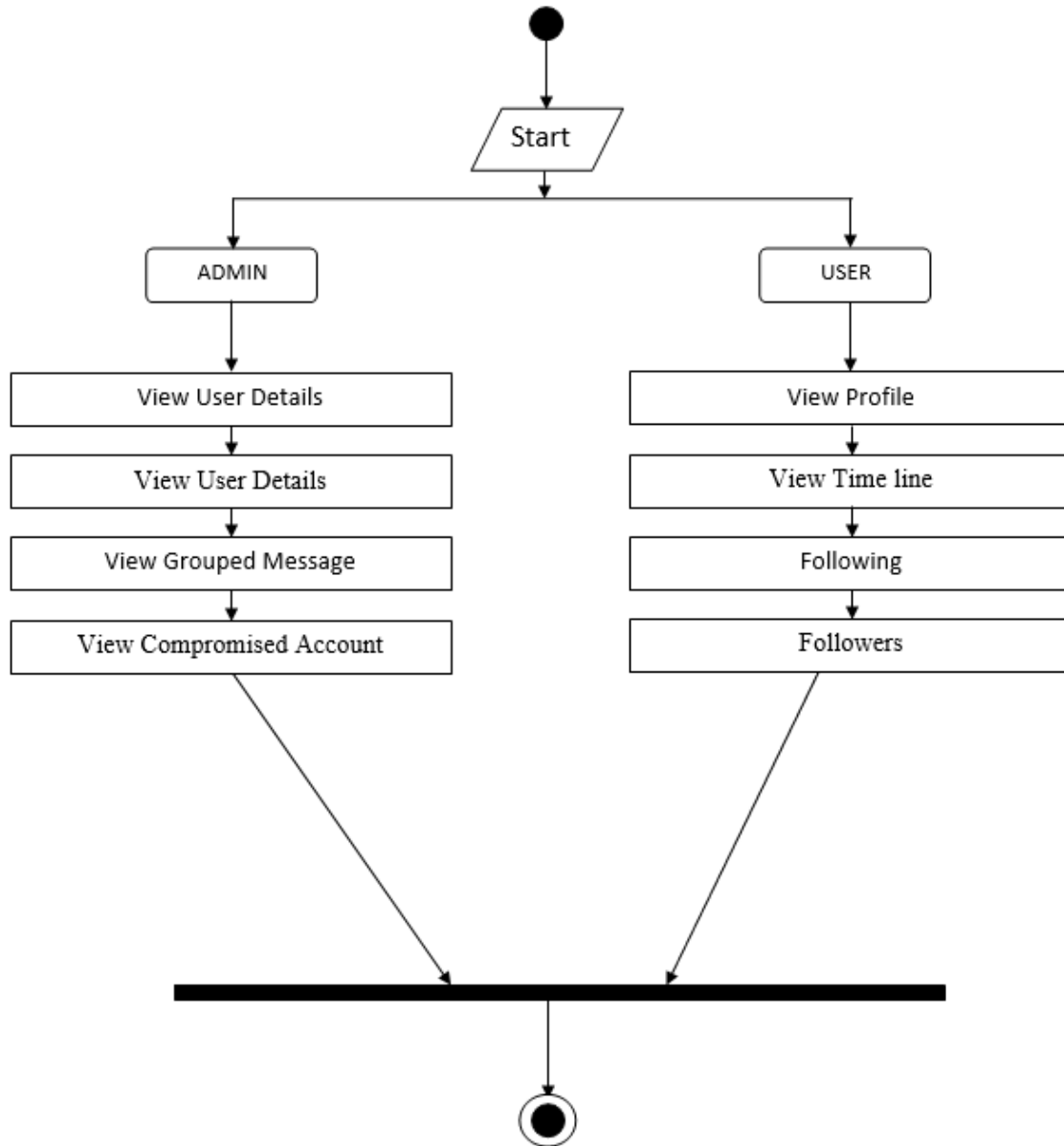
SEQUENCE DIAGRAM:

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 diagrams.



ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling 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.



SYSTEM STUDY

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

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.

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.

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.

SYSTEM 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.

TYPES OF TESTS

Unit testing

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.

Integration testing

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.

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.

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.

White Box Testing

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

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.

6.1 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.

6.2 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.

6.3 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 7 SOURCE CODE

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Login</title>
<link rel="stylesheet" href="dist/css/demo.css" />
<link rel="stylesheet" href="dist/css/style3.css" />
</head>
<body>
<div class="container">

</div>
<div id="container_demo" >

<div id="wrapper">
<div id="login" class="animate form">
<form id="frm1" method="post" action="mainpage.php" autocomplete="on">
<h1>Log in</h1>
<p>
<label for="username" class="uname" > Username</label>
<input id="username" name="txt" type="text" placeholder="Username"
required="required"/>
</p>
<p>
<label for="password" class="youpasswd" >Password </label>
<input id="password" name="pwd" type="password" placeholder="Password"
required="required" />
</p>

<p class="login button">
<input type="submit" name="log" value="Login" />
```

```
<input type="submit" name="can" value="Cancel"/>
</p>
<p class="change_link">
```

```
us for new Registration</a>
```

```
Not a member yet ?
<a href="reg.php">Join
```

```
</p>
```

```
</form>
</div>
```

```
</div>
</div>
```

```
</body>
</html>
```

Admin Page

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Login</title>
<link rel="stylesheet" href="../dist/css/demo.css" />
<link rel="stylesheet" href="../dist/css/style3.css" />
</head>
<body>
<div class="container">

</div>
<div id="container_demo" >

<div id="wrapper">
<div id="login" class="animate form">
<form id="frm1" method="post" action="" autocomplete="on">
<h1>Admin Log in</h1>
<p>
```

```

<label for="username" class="uname" > Username</label>
<input id="username" name="username" type="text" placeholder="Username"
required="required"/>
</p>
<p>
<label for="password" class="youpasswd" >Password </label>
<input id="password" name="password" type="password"
placeholder="Password" required="required" />
</p>

```

```

<p class="login button">
<input type="submit" name="submit" value="Login" />
<input type="reset" name="can" value="Reset"/>
</p>

```

```

<p style='color:red;'>
<?php
if(isset($_GET['error']))
{
    echo "Invalid
}
?>
</p>

```

Credentials";

```

</form>
</div>

```

```

</div>
</div>

```

```

</body>
</html>
<?php
include '../db.php';

```

```

if(isset($_POST['submit']))
{
    $uname = $_POST['username'];
    $pword = md5($_POST['password']);
    //$pword = md5(hash("sha256",$pword));
    $sql ='SELECT * FROM admin WHERE username = "'.$uname.'" AND
password = "'.$pword.'";

```

```

    $results = mysql_query($sql,$conn);
    if ($results) {
        $num_rows = mysql_num_rows($results);

        if ($num_rows> 0) {

session_start();
$_SESSION['user'] = $uname;
header ("Location: dashboard.php");
        }
    else {

$errorMessage= "Error logging on";
header('Location: index.php?error=1');
    }

}
else {

$errorMessage = "Invalid credential";
header('Location: index.php?error=2');

}

}

?>

```

Delete Page

```

<?php

if(isset($_GET['d']))
{
include "../db.php";
$id=$_GET['d'];
$uid = $_GET['uid'];
$sql = "DELETE FROM `share` WHERE id='$id'";
mysql_query($sql);
$sql1 = "INSERT INTO `compa`(`uid`) VALUES ('$uid ')" ;
mysql_query($sql1);
}
header ('location: http://localhost/profile/admin/data.php?s=1');

?>

```

```

Data
<?php
session_start();
include '../db.php';
if(isset($_SESSION['user']))
{
?>
<html>

<head>
<title>Towards Detecting Compromised Accounts on Social Networks</title>
<meta name="description" content="Towards Detecting Compromised Accounts
on Social Networks" />
<meta name="keywords" content="Towards Detecting Compromised Accounts
on Social Networks" />
<meta http-equiv="content-type" content="text/html; charset=windows-1252" />
<link rel="stylesheet" type="text/css" href="style/style.css" title="style" />
<style>

td{text-align:center !important; padding-left:5px !important; padding-right:5px
!important;}
td a {color:red}
</style>
</head>

<body>
<div id="main">
<div id="header">
<div id="logo">
<div id="logo_text">
<!-- class="logo_colour", allows you to change the colour of the text -->
<h1><a href="index.php">Towards Detecting Compromised Accounts on Social
Networks</a></h1>

</div>
</div>
<div id="menubar">
<ul id="menu">
<!-- put class="selected" in the li tag for the selected page - to highlight which
page you're on -->
<li><a href="dashboard.php">Users</a></li>
<li><a href="data.php">Posts</a></li>
        <li><a href="cau.php">COMPA Users</a></li>
        <li><a href="logout.php">Logout</a></li>
</ul>

```

```

</div>
</div>
<div id="site_content">

<div id="content">
<!-- insert the page content here -->
<h1>POSTED IMAGE &nbsp;&nbsp;&nbsp;&nbsp;<?php
if(isset($_GET['s'])){echo "<span style='color:red; border:1px solid
red;padding:5px;'>Malicious Content has been Deleted</span>";}
?></h1>
<?php

$sql = 'SELECT * FROM `share`;

$result_new = mysql_query($sql);
?>
<table cellpadding='20px'>
<tr>
<th>Image</th>
<th>Shared By</th>
<th>Action</th>
</tr>
<?php
while($row=mysql_fetch_array($result_new))
{?>
    <tr>
    <td><imgsrc="..<?php echo $row['pic'] ?>" height='100px'
width='100px'></td>
    <td><i><?php echo $row['username']; ?></i></td>
    <td><a href='delete.php?d=<?php echo $row['id'] ?>&uid=<?php
echo $row['username']; ?>'>Delete Malicious Content</a></td>
    </tr>
<?php
}

?>
</table>

</div>
</div>
<div id="footer">
    Towards Detecting Compromised Accounts on Social Networks
</div>
</div>
</body>

```

```

</html>
<?php
}
else
{
    header ('location: http://localhost/profile/admin/index.php');
}

?>

```

Block
<?php

```

if(isset($_GET['q']))
{
    include "../db.php";
    $uid=$_GET['q'];

    $sql = "UPDATE `user` SET `active`='0' WHERE id='$uid'";
    mysql_query($sql);
}
header ('location: http://localhost/profile/admin/dashboard.php?s=1');

?>

```

CONCLUSION

In this paper, we presented COMPA, a system to detect compromised accounts on social networks. COMPA uses statistical models to characterize the behavior of social network users, and leverages anomaly detection techniques to identify sudden changes in their behavior. The results show that our approach can reliably detect compromises affecting high-profile social network accounts, and can detect compromises of regular accounts, whose behavior is typically more variable, by aggregating together similar malicious messages.

REFERENCE

- Manuel Egeley, Gianluca Stringhini, Christopher Kruegel, and Giovanni Vigna, "Towards Detecting Compromised Accounts on Social Networks", **IEEE Transactions on Dependable and Secure Computing**, 2017.