KEYSTROKE DYNAMICS FOR MOBILE DEVICES –
DATA COLLECTION

A Thesis
Presented to the
Faculty of
San Diego State University

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
in
Computer Science

by
Ritesh Kantilal Dedhia
Summer 2011
SAN DIEGO STATE UNIVERSITY

The Undersigned Faculty Committee Approves the

Thesis of Ritesh Kantilal Dedhia:

Keystroke Dynamics for Mobile Devices – Data Collection

Joseph Lewis, Chair
Department of Computer Science

Kris Stewart
Department of Computer

J. Carmelo Interlando
Department of Science Mathematics and Statistics

Peter Bartoli
Department of Computer Science

April 26, 2011
Approval Date
Copyright © 2011
by
Ritesh Kantilal Dedhia
All Rights Reserved
DEDICATION

I dedicate this thesis work to my father Mr. Kantilal S. Dedhia and my mother Mrs. Lata K. Dedhia for their continuous encouragement. This would not have been possible without their unconditional love and support.
ABSTRACT OF THE THESIS

Keystroke Dynamics for Mobile Devices – Data Collection
by
Ritesh Kantilal Dedhia
Master of Science in Computer Science
San Diego State University, 2011

In this day and age, mobile devices are developed to serve various functions; storing sensitive information such as passwords, bank credentials, and important data. As a consequence, authenticating users for mobile devices has become an important issue. However, unlike desktops and notebook computers, a four digit personal identification number (PIN), is often adopted as the sole security mechanism for mobile devices. Due to their limited length, PINs are vulnerable to surfing and systematic trial-and-error attacks.

Additionally, the development of the mobile system is moving forward to touch screen system for user-friendly and quick access mechanism. This thesis project is one of the two related thesis projects that demonstrate the effectiveness of user authentication using keystroke dynamics-based authentication on mobile devices. Key down, key up times and the key ASCII codes are the inputs captured from the user typing. Four features, (key code, two keystroke latencies, and key duration) are analyzed while capturing samples from the user and stored in the database in a format; which is easily used by a related thesis project. The stored samples are then compared with the current sample using an algorithm to authenticate users. Google Android is used as a development platform for implementation.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>INTRODUCTION</td>
</tr>
<tr>
<td>2</td>
<td>BACKGROUND AND LITERATURE</td>
</tr>
<tr>
<td>2.1 Common Biometric Characteristics</td>
<td>4</td>
</tr>
<tr>
<td>2.2 Biometric Systems</td>
<td>4</td>
</tr>
<tr>
<td>2.3 Methods of Authentication</td>
<td>5</td>
</tr>
<tr>
<td>2.4 Different Types of Biometric Techniques</td>
<td>6</td>
</tr>
<tr>
<td>2.4.1 Finger Print Recognition</td>
<td>6</td>
</tr>
<tr>
<td>2.4.1.1 Image Capture</td>
<td>6</td>
</tr>
<tr>
<td>2.4.1.2 Fingerprint Minutiae</td>
<td>6</td>
</tr>
<tr>
<td>2.4.1.3 Minutiae Extraction</td>
<td>7</td>
</tr>
<tr>
<td>2.4.1.4 Storage</td>
<td>7</td>
</tr>
<tr>
<td>2.4.1.5 Matching</td>
<td>8</td>
</tr>
<tr>
<td>2.4.1.6 Sweeping Technique</td>
<td>8</td>
</tr>
<tr>
<td>2.4.2 Face Recognition</td>
<td>8</td>
</tr>
<tr>
<td>2.4.3 Iris Recognition</td>
<td>10</td>
</tr>
<tr>
<td>2.4.4 Hand Geometry</td>
<td>13</td>
</tr>
<tr>
<td>2.4.5 Voice Biometrics</td>
<td>13</td>
</tr>
<tr>
<td>2.4.6 Signature Verification</td>
<td>13</td>
</tr>
<tr>
<td>2.4.7 Keystroke Dynamics</td>
<td>14</td>
</tr>
<tr>
<td>2.5 Other Biometric Techniques</td>
<td>16</td>
</tr>
<tr>
<td>2.6. Working of Biometrics</td>
<td>16</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Block diagram of a biometric system.</td>
<td>5</td>
</tr>
<tr>
<td>2.2</td>
<td>Fingerprint image.</td>
<td>7</td>
</tr>
<tr>
<td>3.1</td>
<td>Android architecture.</td>
<td>23</td>
</tr>
<tr>
<td>3.2</td>
<td>Activity.</td>
<td>24</td>
</tr>
<tr>
<td>3.3</td>
<td>Conversion from .java to .dex file.</td>
<td>26</td>
</tr>
<tr>
<td>4.1</td>
<td>Registration screen from the Keystroke Dynamics Application on December 15, 2010.</td>
<td>31</td>
</tr>
<tr>
<td>4.2</td>
<td>Registration screen – success from the Keystroke Dynamics Application on December 15, 2010.</td>
<td>31</td>
</tr>
<tr>
<td>4.3</td>
<td>Collecting samples from the Keystroke Dynamics Application on December 15, 2010.</td>
<td>32</td>
</tr>
<tr>
<td>4.4</td>
<td>Clear samples from the Keystroke Dynamics Application on December 15, 2010.</td>
<td>33</td>
</tr>
<tr>
<td>4.5</td>
<td>Login screen from the Keystroke Dynamics Application on December 15, 2010.</td>
<td>33</td>
</tr>
</tbody>
</table>
LIST OF ABBREVIATIONS

ADT – Android Development Tools
API – Application Programming Interface
FAR – False Acceptance Rate
FRR – False Rejection Rate
HCI – human computer interface
Hd – Hamming distance
ICA – Independent Component Analysis
IDE – integrated development environments
KD – Keystroke Dynamics
KPCA – Kernel PCA
LDA – Linear Discriminate Analysis
PCA – Principal Component Analysis
PIN – Personal Identification Number
SDK – Software Development Kit
ACKNOWLEDGEMENTS

I would like to take this opportunity to sincerely thank my Thesis Advisors Dr. Joseph Lewis and Mr. Peter Bartoli. I have continuously worked with Mr. Peter Bartoli on this thesis to discuss various approaches and solutions. His support and encouragement were instrumental in achieving the optimum results in this thesis. Dr. Joseph Lewis has also been a vital source of encouragement, discussing the contents of my thesis.

I would also like to thank Dr. Kris Stewart, Prof. Carmelo Interlando for providing valuable suggestions and taking their time to review my thesis, as part of the panel.
CHAPTER 1

INTRODUCTION

At present, mobile devices are used not only make or receive calls, but also provide extraordinary assistance in the field of business, such as; accessing the internet, buying and selling stocks, transferring money, and managing bank account, to say the least. In addition, it can be conjectured that third generation handsets are more advanced in their ability to pay for products using micropayments and digital money. Furthermore, the development of mobile system is moving forward to touch screen system for user-friendly and quick access mechanism. However, the rise in computing mobility is causing a number of security issues, in particular with attackers accessing the data stored on the device. As a consequence, the authentication of users for mobile devices is of concern.

The most popular access security in mobiles is either a password or personal identification number (PIN), a secret-knowledge approach that relies heavily on the user to ensure continued validity. Due to their limited length, PINs are vulnerable to surfing as well as trial-and-error attacks. Therefore, an alternate means of user authentication is required to replace the secret-knowledge based approaches. It is thus appropriate to examine the potential of other fundamental strategies. Among the available techniques, biometric-based authentication is the only one that seems plausible since tokens must also be carried in token-based authentication, along with the device.

Biometrics is not based on what the user knows, or what they carry, but rather, it is based on the user’s unique characteristics. One such biometric that lends itself to a mobile context, due to the keypad already residing on the handset, is Keystroke Dynamics – which authenticates the user by their typing style. The principle behind keystroke dynamics is to extract and analyze the way an individual types as opposed to what the individual types. Features that can be extracted of the keystroke rhythm are; the time when a key is pressed, the time between successive key presses, speed of the keystroke, etc. This technology is relatively cheaper than the fingerprint or retinal scan technology, which requires expensive and extra hardware for data collection, making it ideal for mobile device security.
The main question to answer is whether Keystroke Dynamics can be used as a security mechanism for mobile devices. The effectiveness and accuracy of Keystroke Dynamics needs to be evaluated. The advantages and disadvantages of this biometric technique will be studied.

The thesis ‘Keystroke Dynamics on Mobile Devices’ is divided into two parts: ‘Data Collection’ and ‘Algorithm and Authentication’. My part deals with collection of Keystroke Data such as dwell time, flight time, and row switch time from the touchpad, validating the data and organizing the data into an appropriate format. The other part of the thesis ‘Algorithm and Authentication’ is implemented by Mradul Shrivastava. It involves reading the data from the database, creating an algorithm to authenticate users and perform the testing of the application.

This thesis is organized as follows: Chapter 2 will explain the background related to Biometrics. Chapter 3’s purpose is to describe, the technology used. In Chapter 4, the implementation of Mobile Keystroke Dynamics and results are presented and discussed. In Chapter 5, the database schema is described. Finally, Chapter 6 presents the conclusions and future works.
CHAPTER 2

BACKGROUND AND LITERATURE

A lot of activities are carried out in today’s digital world, which increases the importance of simple and secure system for everyone. The need for secure and personal identification and verification technologies has become a great concern. Protecting Information is the main aim of any organization and individual as it is their greatest asset. Security of this information from threats plays an important role for any organization. Many biometric techniques have been developed to make the system secure. Biometric techniques are a reliable method of recognizing the identity of a person based on physiological or behavior characteristics.

The physical or behavioral traits of human are exploited by the biometric techniques in order to authenticate people. A number of features such as face, fingerprints, hand geometry, iris, retinal, voice, etc. are measured. Biometric authentications are used by a lot of industries such as banking, stock exchange, retailing, defense, health industry. Biometric techniques provide a robust solution to many challenging problems in security. It verifies the identity of an individual based on the measurement and analysis of unique physical and behavioral data. Biometric techniques are increasingly being viewed as the preferred means to confirm an individual’s identity accurately [1].

“Biometrics (ancient Greek: Bios=“life”, metric=“measure”) is the study of methods for uniquely recognizing humans based upon one or more intrinsic physical or behavioral traits” [2]. Biometrics is an automatically measurable, robust and distinctive physical characteristic or personal trait that can be used to identify an individual or verify the claimed identity of an individual [3]. Measurable means that the characteristic or trait can be easily presented to a sensor, located by it, and converted into a quantifiable, digital format. The robustness of a biometric refers to the extent to which the characteristic or trait is subject to significant changes over time. These changes can occur as a result of age, injury, illness, occupational use, or chemical exposure. A highly robust biometric does not change
significantly over time while a less robust biometric will change. Distinctiveness is a measure of the variations or differences in the biometric pattern among the general population.

2.1 COMMON BIOMETRIC CHARACTERISTICS

Biometric characteristics can be divided into two main classes:

1. Physiological
   - Face
   - Fingerprint
   - Hand
   - Iris

2. Behavioral
   - Keystroke
   - Signature
   - Voice

A biometric system is essentially a pattern recognition system which makes a personal identification by determining the authenticity of a specific physiological or behavioral characteristic possessed by the user.

2.2 BIOMETRIC SYSTEMS

The main functions of a biometric system are enrollment and test. The enrollment phase involves storing the information of an individual whereas the test phase involves testing the biometric data with the stored information. The first block (sensor) acts as an interface between the real world and the biometric system; it acquires all the crucial data. Normally it is an image acquisition system which can change according to the characteristics we may want to consider. The pre-processing is performed by the second block which involves removing artifacts from the sensor and enhancing the input (e.g. to remove some noise) by using some kind of normalization, etc. We extract the features which are required in the third block. This step stays important as we need to choose which features to extract and how. The flow of the biometric system can be visualized in Figure 2.1.

After extracting the features, we get a vector of numbers or an image with particular properties which can be used to create a template. A template is a blend of all the
characteristics that we extract from the source. It should normally be small in order to be efficient, but big enough to discriminate clearly. For enrollment purpose, the template is simply stored in a database. During the matching phase, the obtained template is compared with other existing templates, estimating the distance between them using any algorithm (e.g. Hamming distance). The matcher distinguishes whether the obtained template matches with the stored information. (It tells whether the user is authentic or not) [2].

2.3 METHODS OF AUTHENTICATION

**Identification:** In an identification biometric system, the biometric device reads a sample and compares that sample against every record or template in the database. This is a “one-to-many” (1: N) type of comparison. The system either gives the best match or gives a list of possible matches in the order of their likelihood. These types of biometric systems are used when goal is to identify criminals, terrorists particularly through surveillance.

**Verification:** A verification biometric system checks the user against his/her own stored template data. The system takes the user input in the form of username, password or token and compares it against the template in the database. It also takes a biometric sample from the user. It then compares the sample to or against the user-defined template. This is a
“one-to-one” (1:1) type of comparison. The system either finds a match or fails. It does not return a list of possible matches in this case. Verification is commonly used for physical or computer access [3].

2.4 Different Types of Biometric Techniques
The Biometric Techniques are discussed in detail as below.

2.4.1 Finger Print Recognition
A fingerprint based biometric system is a pattern recognition system that recognizes a person by determining the authenticity of his fingerprint [4]. It is based on the fact that each individual has unique fingerprints. It is one of the most widely used and actively studied biometric technologies. The uniqueness of fingerprints has been acknowledged and exploited by law enforcement for more than 100 years. Fingerprint recognition is being combined with powerful microprocessors and pattern matching software to open new applications in commercial market such as computers, banking, cars, and cellular phones. Wherever we use a key or a password, it can be replaced by our fingerprint.

Fingerprint recognition can be divided into two distinct phases: enrollment and recognition. Enrollment involves image capture, signature extraction and storage.

2.4.1.1 Image Capture
Image capture consists of obtaining an image of the fingerprint. It includes reading the fingerprint using fingerprint scanner. The most important part of a fingerprint scanner is the sensor (or sensing element) which is the component where the fingerprint image is formed. Almost all the existing sensors belong to one of the three families:

- Optical
- Solid-state
- Ultrasound

2.4.1.2 Fingerprint Minutiae
Several techniques have been proposed to perform physical sensing without using optics: Pressure, capacitance, thermal, switches, capacitance—flat panel, RF field.
As shown in Figure 2.2, the flows of the black lines are called ridges. The space between the ridges is called a valley. Ridges and valley often run parallel; sometimes they bifurcate and sometimes they terminate at an ending point. These points are called minutiae and form significant information in the classification of an automatic fingerprint system [5].

![Fingerprint image](image)

**Figure 2.2.** Fingerprint image.

### 2.4.1.3 MINUTIAE EXTRACTION

Most of the proposed methods require the fingerprint gray scale image to be converted into a binary image followed by thinning stage which allows for the ridge line thickness to be reduced to one pixel. Maio and Maltoni [4] proposed a direct gray-scale minutiae extraction technique, whose basic idea is to track the ridge lines in the gray-scale image. The ridge line extraction algorithm attempts to locate, at each step, a local maximum relative to a section orthogonal to the ridge direction. By connecting the consecutive maxima, a polygonal approximation of the ridge line can be obtained.

### 2.4.1.4 STORAGE

Once the template is obtained, it must be stored for further recognition. Signature can be stored almost anywhere as it is very small in size, even on a hard drive or in a smart card. The important point is to store the template so that it can be easily retrieved later for recognition.
2.4.1.5 MATCHING

Matching consists of comparing the reference signature/template, stored during enrollment and the live template obtained from the user attempting to be recognized. Biometrics distinguishes between two types of matching: ‘1 to 1’ and ‘1 to many’, depending on the number of reference signatures as well as the storage method. ‘1 to many’ is known as identification. A police application is a typical example, where you attempt to find the match for one fingerprint image among a huge database. ‘One to one’ is known as verification. In this case, you want to prove that you are who you say you are. You may state your identity, for instance, by typing an identification number or inserting a smart card. Your fingerprint is then checked against the reference signature selected with the identification number or previously stored in the smart card.

2.4.1.6 SWEEPING TECHNIQUE

In order to save the cost of silicon chips used for fingerprint imaging, the sweeping technique has been developed, which is patented by Atmel. The user sweeps his/her finger over a thermal, rectangular shaped array. To enable image reconstruction without calculating finger speed, the method uses several lines per “slice” and searches identical pixels between each “slice”. The sweeping technique allows for a reduction of silicon area (and cost) by a factor of five compared to using a square-imaging array.

2.4.2 Face Recognition

Face recognition is a biometric technique which can automatically identify a person by the face. It analyzes special features in the face such as distance between the eyes, width of the nose, position of cheekbones, jaw line, chin, unique shape, pattern, etc. Measurement of the eyes, nose, mouth and other facial features are involved in these systems. Facial expressions as well as mouth and lip movement are measured to increase the accuracy. Face recognition captures characteristics from a digital image or a video frame from a video source and translates unique characteristics into a set of numbers. The collected data from the face are combined in a single unit which uniquely identifies each person. The entire face of the person is taken into consideration or the different parts of the face are taken into consideration for the identity of a person.
Face recognition has a lot of applications in the field of security. Facial recognition sets aside from other biometrics since it can be used for surveillance purposes. For example, public safety authorities want to locate certain individuals such as wanted criminals, suspected terrorists, and missing children. Facial recognition may have the potential to help the authorities with this mission. It can also be used as a replacement to the ATM pin, computer passwords and so on. Additional applications include automated crowd surveillance, access control, face reconstruction, design of human computer interface (HCI), multimedia communication, driver’s license and voter registration.

Collateral information such as race, age, gender, facial expression may be used for narrowing the search. It also involves segmentation of faces from cluttered scenes and feature extraction from the face region. Face recognition can be used for identification as well as verification. In identification, the input to the system is an unknown face and the system reports back the determined identity from a database of known individuals, whereas in verification, the system needs to confirm or reject the claimed identity of the input face.

Face Recognition can be classified as Appearance based and Model based. Appearance based are further classified as Linear which includes Principal Component Analysis (PCA), Independent Component Analysis (ICA) and Linear Discriminate Analysis (LDA) and Non-Linear which includes Kernel PCA (KPCA), ISOMAP, LLE, etc. Model based Face recognition is further classified as 2D and 3D. 2D includes Elastic Bunch Graph and Active Appearance Model.

Face recognition research has been focused on detecting individual features such as the eyes, nose, mouth and head outline and defining a face model by the position, size and relationships among these features. The most popular method is called PCA, which is commonly referred to as the eigenface method. PCA is used for dimensionality reduction to find the vectors which best account for the distribution of face images within the entire image space. PCA has also been combined with neural networks and local feature analysis in efforts to enhance its performance. PCA and ICA construct the face space without using the face class information. The whole face training data is taken as a whole. In LPA, the goal is to find an efficient or interesting way to represent the face vector space. Swets and Weng [6] first proposed PCA plus LDA. Their approach composed of two processes, the PCA process followed by the LDA process. They reported a peak recognition rate of more than 90% [7].
The model based face recognition scheme constructs a model of human face, which is able to capture the facial variations. The model-based scheme usually contains three steps: (1) Constructing the model; (2) Fitting the model to the given face image; (3) Using the parameters of the fitted model as the feature vector to calculate the similarity between the query face and prototype faces in the database to perform the recognition.

In Feature-based Elastic Bunch Graph Matching, faces are represented as graphs, with nodes positioned at fiducial points (such as the eyes, the tip of the nose, some contour points, etc.) and edges labeled with 2-D distance vectors. To identify a new face, the face graph is positioned on the face image using elastic bunch graph matching. The goal of Elastic graph matching is to find the fiducial points on a query image and thus to extract from the image a graph which maximizes the graph similarity function [8].

3D face recognition uses the 3D geometry of the human face. 3D face recognition can achieve a lot higher accuracy than 2D face recognition. Pitfalls of 2D face recognition such as different facial expression, head orientation and change in lighting can be avoided using 3D. Recent solutions have implemented depth perception by projecting a grid onto the face and integrating video capture of it into a high resolution 3D model [8].

Face Recognition faces special challenges in case of pose variation, illumination conditions, scale variability, images taken years apart, glasses, moustaches, beards, low quality image acquisition, partially occluded faces, etc. Initially, face recognition systems focused on still images. However, recently image sequences have gained much attention, although nearly all systems apply still-image face recognition techniques to individual frames. Video-based face recognition provides several advantages over still image based face recognition such as good frames can be selected on which to perform the recognition stage. It allows tracking of images such that facial expressions and pose variations can be compensated for, resulting in improving recognition.

### 2.4.3 Iris Recognition

Iris recognition is a method of biometric authentication that uses pattern-recognition techniques based on high resolution images of the irises of an individual’s eyes. The annular area between the pupil and the white sclera in the eye is called the Iris. It has a rich texture based on interlacing features, which is called the texture of the iris. This texture is well-
known to provide a signature that is unique to each individual [9]. Retina scan technology maps the capillary pattern of the retina, a thin nerve on the back of the eye. It analyses the iris of the eye, which is the colored ring of tissue that surrounds the pupil of the eye. Retina scanning captures unique pattern of blood vessels whereas the iris scanning captures the iris. Retina scan is used in military and government organization for authentication in high-end security applications to control access. The human iris has a unique pattern which remains unchanged throughout one’s lifetime.

Iris recognition uses camera technology, with exquisite infrared illumination which reduces specular reflection from the convex cornea in order to create images of the detail-rich, intricate structures of the iris. When converted into digital templates, these images provide mathematical representations of the iris that yield distinct positive identification of an individual.

General iris recognition system consists of the following basic modules:

- Image acquisition, iris location, and pre-processing,
- Iris texture feature extraction and signature encoding, and
- Iris signature matching for recognition or verification. [10]

There has been a lot of research in the field of Iris Recognition. We will discuss the few important ones; starting with the best known Daugman Algorithm. The Daugman algorithm first locates the pupillary and limbic boundaries of the iris using an integrodifferential operator which finds the circles in the image where the intensity is changing most rapidly with respect to changes in the radius. Once located, the iris image is converted to a Cartesian form by projecting it onto a dimensionless pseudo-polar coordinate system. The iris features are encoded and a signature is created using a 2-D complex-valued Gabor filter, where the real and imaginary parts of each outcome are assigned a value of 0 or one according to whether they are negative or positive, i.e. only the quadrant of the phase is encoded. Finally, two images are said to be independent if their fractional Hamming distance (Hd) is above a certain threshold, about .33, otherwise they are a match. Here, Hd equals number of mismatching bits divided by number of compared bits. When applied to a very large database, the Daugman algorithm has been essentially error-free [9, 11].

C. Tisse, L. Martin, L. Torres, and M. Robert [12] presented a modification of Daugman’s algorithm, with two major differences. The two innovations were in the iris
location and feature extraction stages. The use of dimensionless polar coordinates and Hamming distance remain the same. To locate the iris, their algorithm applies a gradient decomposed Hough Transform to find the approximate center of the pupil, and then applies the integrodifferential operator, as in Daugman’s algorithm, to find the precise locations of the iris boundaries. This combined approach has the advantage of avoiding errors due to specular reflection in the images. The Hilbert Transform is used to create an analytic image in the feature extraction and encoding step, whose output is then encoded as an emergent frequency vector and an instantaneous phase. This approach has an advantage of being computationally efficient. The Tisse algorithm has reportedly been successful when tested on a database of real iris images [13].

A key problem with existing approaches to iris signature acquisition is the limited depth-of-field of traditional imaging systems. In order to achieve reasonable lighting levels and exposure times, the optical system must have a high numerical aperture and a corresponding low F-number. Unfortunately, a high numerical aperture results in a corresponding small depth-of-field. Commercial iris recognition implementations typically require the user to move his or her head back and forth with respect to the camera until the focus quality is good enough to provide a sufficiently high contrast iris signature. Some implementations rely on audio cues and others rely on visual cues to let the user know when the iris is at an appropriate distance from the camera. The process can be time-consuming and is an obstacle to acceptance of the process in daily use [14].

Iris Recognition like any other biometric identification technology is also not a completely solved problem due to the problem of live-tissue verification. The biometric identification is only reliable if it is captured from a live body part of a human, then compared unlike any manufactured template. A lot of commercially available iris-recognition systems do not validate the real human face and even a high quality photograph can easily fool these systems. Such devices are not applicable for unsupervised applications, such as door access-control systems. Although in supervised applications, the problem of live-tissue verification is less of a concern (e.g., immigration control), where a human operator supervises the process of taking the picture.

Some of the below methods can be used against the use of fake eyes and iris such as:
- Changing ambient lighting during the identification (switching on a bright lamp), such that the pupillary reflex can be verified and the iris image can be recorded at several different pupil diameters.
- Analyzing the 2D spatial frequency spectrum of the iris image for the peaks caused by the printer dither patterns.
- Analyzing the temporal frequency spectrum of the image for the peaks caused by computer displays. [15]

2.4.4 Hand Geometry

Hand geometry is a biometric technique which captures the physical characteristics of a user’s hand and fingers. The bifurcations or branches which are made by the ridges and finger image ridge endings are analyzed. The length, width, thickness and surface area of an individual’s hands are recorded by the system. This biometric technique is easy to use and is widely used due to its cost factor, which is relatively low. It is normally used in applications such as access control and attendance. A 3D image of the hand is captured by the camera, then a verification template is created and stored in the database by taking samples from the user. At the time of verification of a person, the current template is compared with the stored sample templates. Depending on the result of the match between the sample and the templates, the user is granted access or denied access [1].

2.4.5 Voice Biometrics

Voice biometrics is another biometric technique which uses the person’s voice to verify or identify the person. It can be used for verification as well as identification purpose. The unique characteristics of a person can be analyzed using just a microphone with specific software. Telephone-based applications mostly use this biometric technique. In order to record samples, the user is required to speak a given phrase into the microphone or telephone handset. The sample is analyzed by the system for numerous characteristics, including pitch, tone, and shape of larynx and then creates a template. Voice verification is easy to use and one of the least intrusive of all biometric methods [1].

2.4.6 Signature Verification

Signature Verification is a process used to recognize an individual’s handwritten signature by analyzing speed. Acceleration rate, stroke length and pressure applied during the
Dynamic signature verification uses behavioral biometrics of a handwritten signature to confirm the identity of a person. This can be achieved by analyzing the shape, speed, stroke, and pen pressure and timing information during the act of signing [1]. There are different ways to capture data for analysis, i.e. a special pen can be used to recognize and analyze different movements when writing a signature, the data will then be captured within the pen. Information can also be captured within a special tablet that measures time, pressure, acceleration and the duration the pen touches it. As the user writes on the tablet, the movement of the pen generates sound against paper, which is used for verification [17].

The simple signature comparison only takes into account what the signature looks like. But with dynamic signature verification, it is not the shape or look of the signature that is meaningful, it is the changes in speed, pressure and timing that occur during the act of signing, thus making it virtually impossible to duplicate those features. An individual’s signature can change over time, however, which can result in the system not recognizing authorized users. Devices which enable dynamic signature verification store the behavioral factors and the captured signature image itself for future comparison in their database. These devices account changes in one’s signature over time by recording the time and the dynamic features each time a person uses the system [18]. When the user signs his name on an electronic pad, rather than merely comparing signatures, the device instead compares the direction, speed and pressure of the writing instrument as it moves across the pad. Such systems are mostly used in Banks.

### 2.4.7 Keystroke Dynamics

Keystroke Dynamics is a behavioral biometric technique that uses the rhythm and manner in which an individual types characters on a keyboard. The keystroke rhythms of a user generate a unique biometric template of the individual’s typing pattern for authentication. The key measurements such as Dwell time (how long a key is pressed) and Flight time (how long it takes to move from one key to another) are used to determine the unique typing rhythm of a user. The recorded data is then processed through a unique algorithm which determines the user pattern for future comparison. Keystroke dynamics is mostly used for identification purposes.
Every individual has a different typing style. Some users type fast while some users type very slowly. In order to identify if the user is valid, we can compare the typing speed of the user and check if the speed matches. Thus an authentic user who types 30 words per minute can easily be identified from an impostor who types 70 words per minute. The amount of time a user presses a key and the time required by a user to switch between keys is also different. Most people have specific letters that take them longer to find or get to than their average seek-time over all letters, but those letters may vary dramatically for different people. Right-handed people are sometimes faster in getting to keys they hit with their right hand fingers than they are with their left hand fingers and similarly with left-hand people. Index fingers may be characteristically faster than other fingers. These key differences of an individual which differentiates himself from other are used in Keystroke Dynamics to authenticate a user.

As the person types, the Keystroke Dynamics (KD) application collects the time each key is pressed down and the cycle time between one key-down and the next. For verification purposes a known verification string is typically typed (e.g. account ID and password). Once the verification string is entered, it is processed through an algorithm that compares the person’s typing behavior to a sample collected previously. The output of the comparison is a score. In most cases some samples are collected from the user before verifying the user’s identity. If the score falls within a range defined as acceptable, and the password entered is correct, the user is authenticated and verified—access is granted. Every time a user successfully logins, the data is stored as a sample for future comparisons. Gradually if the typing style of the user changes, the KD application still verifies the user correctly due to its Artificial Intelligence.

There are definitely a few advantages of Keystroke Dynamics over other biometric techniques. Since Keystroke Dynamics is used along with a PIN or a password to verify identity, it is very resistant to counterfeiting. Keystroke Dynamics is reliable as it just uses regular keyboard unlike fingerprint scanner which after aged increases error rates. It is easily accepted by users unlike fingerprint and retina scanners where the user doesn’t have to touch or be touched by any special device. Further, no information about the user (fingerprint, retinal print, voice print, etc.) is stored. It is also cost effective as extra hardware such as special sensors is not required.
Compared with other biometric techniques, Keystroke Dynamics does not require any special equipment or hardware. It can be implemented using the same keyboard, which is used with a desktop or a laptop. This helps saving extra cost required to implement this biometrics. Like other biometric techniques, Keystroke Dynamics also has some shortcomings since the user’s typing might not be always consistent which will result in more False Rejection Rates (FRR). Increasing the acceptance threshold for FRR will lead to more False Acceptance Rates (FAR).

2.5 Other Biometric Techniques

Few other biometric techniques are as listed below:

- Vein patterns
- Analyzing Nail identification: Analyses the tracks in the nails.
- DNA patterns
- Sweat pore analysis: The pores on a finger are analyzed.
- Ear recognition: Shape and size for each person is unique.
- Odor detection
- Walking recognition

2.6. Working of Biometrics

All biometric systems works in a four-stage process that consists of the following steps:

- Capture: Biometric characteristics such as fingerprint, voice etc. is captured by the biometric system.
- Extraction: The captured data sample is extracted into a unique template. The unique features after extraction are then converted into digital biometric code. This sample is then stored as the biometric template for that individual.
- Comparison: The template is then compared with a new sample.
- Match/non-match: The new sample when compared with the stored template is evaluated by the system into a match or a non-match. It eventually grants or refuses access to the user. [1]

2.7. Biometric Performance Measures

Performance of a Biometric System can be measured using FAR and FRR detailed as follows:
The false acceptance rate (FAR) is the measure of the likelihood that the biometric security system will incorrectly accept an access attempt by an unauthorized user. FAR of a system is calculated as the ratio of the number of false acceptances divided by the number of identification attempts [19].

The false rejection rate (FRR) is the measure of the likelihood that the biometric security system will incorrectly reject an access attempt by an authorized user. FRR of a system is calculated as the ratio of the number of false rejections divided by the number of identification attempts [20].

FAR and FRR must be as low as possible, but both are antagonists and part of an intricate balancing act. If you make the system harder to enter for an impostor – reducing the FAR – you also make the system harder to enter for a true enrollee by raising the FRR. The same occurs in reverse, too.

2.8 APPLICATION OF BIOMETRIC TECHNIQUES

As an emerging technology, biometrics has been widely used in different organization for the security purpose. Preventing unauthorized access to ATMs, workstations, cellular phones, desktop PCs, and computer networks are some of the uses of Biometrics. Telephonic and Internet transactions (e-commerce and e-banking) also use different Biometric techniques. Increasing security threats have forced many countries to start using biometrics for border control and national ID cards [1]. The applications where biometric technique has its presence are:

- Banking
- Physical access control of buildings, areas, doors and cars.
- Electronic access to services (e-banking, e-commerce)
- Border control
- Airport security
- Cyber security
- Voice Recognition (Tele-banking)
- Prison visitor monitoring system.
- Voting System

2.9. ADVANTAGE OF BIOMETRICS

Some of the advantages of Biometric Systems are as follows:

- Increase security – additional security at low cost.
- Reduce fraud by employing hard-to-forge technologies and materials. E.g. minimize the opportunity for ID fraud, buddy punching.
- Physiological biometric systems can eliminate problems caused by lost IDs or forgotten passwords. E.g. prevent unauthorized use of lost or stolen ID cards.
- Reduce the cost involved in password administration.
- Offer significant cost savings or increasing ROI in areas such as Loss Prevention or Time & Attendance.
- An Individual can be linked indubitably to a transaction or event [21].
CHAPTER 3

TECHNOLOGY

This chapter focuses on the technology used to develop the software and briefly discusses about the supporting software, tools and integrated development environments (IDE). Chapter starts with the requirements of the project followed by the description about language used for building the project.

3.1 REQUIREMENTS

The prime objective of building keystroke dynamics is to analyze its accuracy on mobiles phones. The positive behavior of this technique may result as a login authentication provider for mobiles phones and other portable PC. The requirements gathered are enlisted below.

- The main objective of the thesis is to check whether this technique can be implemented in future for mobile phone. Hence, the technology used for this should be latest and the next big thing in future. Android OS with firmware 1.6 or above has been chosen for the development.
- The language used in Android OS is Java. Hence, Java SDK is also required to develop software on Android OS.
- Database used for mobile devices is primarily SQLite. Unlike most other SQL databases, SQLite does not have a separate server process. SQLite reads and writes directly to ordinary disk files.
- All mobile devices having Android OS installed comes with touch pad. Thus touch keyboard should be used for giving inputs to the software.

3.2 JAVA

The Keystroke Dynamics software is built using Java Software Development Kit 6 on Android OS. The integrated development environment used for the writing Java classes is Eclipse Galileo. The reasons behind choosing Java over other software languages include the following.

- The Android user space is largely dominated by Java technologies that run on top of Google’s custom Dalvik Java virtual machine
- Java is simple, easy to implement and object oriented.
- Java provides high performance using its very large set of application programming interfaces (APIs).
- Java is robust and secure.
- Java can provide multi-threaded programming so that the program execution is faster and it is dynamic.
- Java is platform independent, architecturally neutral and highly interpretable.

An object oriented programming language is one which lets you create objects. An object is an entity which drives the class attributes and functions to which it belongs. An object oriented model is a collection of interacting objects which is different from conventional programming. Java is object oriented because it focuses on creating objects and making them work together. The process of creating an object is known as instantiation. Java features all of the object oriented concepts mentioned below.

- Polymorphism: A single method can generate different set of results when passed with different set of arguments.
- Inheritance: Classes are arranged hierarchically and child class can access methods and attributes of its parent class.
- Data encapsulation: Attributes, variables and methods of a particular class are differentiated based on their role in the programming model such as public, private, protected.

Things like these make programming very loosely coupled so that the complexity is reduced and the programming models become highly independent and modular.

### 3.2.1 Simplicity

Java compiler automatically creates the Java compiled classes into machine readable byte-code. The most important feature of Java which makes it very simple is its ability to handle automatic memory management. Java uses automatic garbage collection when an object is destroyed to release the memory unlike C++ where programmer is responsible for freeing the memory associated with the deleted object.

### 3.2.2 Robustness

A robust programming language is very stable, secure and does not fall prey to third party trapdoors. Hence it is very reliable. This is because Java is highly supported language,
intended for use in networked environment. No programming language can really assure full-proof reliability but there aren’t much security holes in Java. An example is a bad Java program will never crash your computer unlike a C program. Java is dynamic in a sense that Java puts lot of emphasis on runtime error checking and eliminating situations which are error prone.

3.2.3 Multi-Threading

A multithreaded program divides any process into several threads. A thread is a smallest unit of program execution. These individual threads run in parallel to allow faster execution of a program and increase the program execution speed. Java has a separate API dedicated for multithreaded programming which has been smoothly integrated into it unlike C++ where operating system specific procedures have to be called in order to enable multithreading.

3.3 ANDROID

Android is an open-source platform developed by Google and the Open Handset Alliance on which interesting and powerful new applications can be quickly developed and distributed to many mobile device users. Some of Android-based devices are Motorola Droid, HTC Droid ERIS, and Google Nexus. Android is a software stack for mobile devices that includes an operating system, middleware and key applications. The Android SDK provides the tools and APIs necessary to begin developing applications on the Android platform using the Java programming language. Android was built from the ground-up to enable developers to create compelling mobile applications that take full advantage of all a handset has to offer. It was built to be truly open. For example, an application can call upon any of the phone’s core functionality such as making calls, sending text messages, or using the camera, allowing developers to create richer and more cohesive experiences for users. Android is built on the open Linux Kernel. Furthermore, it utilizes a custom virtual machine that was designed to optimize memory and hardware resources in a mobile environment. Android is open source; it can be liberally extended to incorporate new cutting edge technologies as they emerge. Android Development Tools (ADT) is a plug-in for the Eclipse IDE that is designed to give you integrated environment to build Android applications. ADT
extends the capabilities of Eclipse to let you quickly set up new Android projects, create an application UI, add components based on the Android Framework API, debug your applications using the Android SDK tools, and even export signed/unsigned APKs to distribute your application.

Android applications written in Java, compiled Java code that is packaged into an Android package known as an .apk file. Bundling application data in this way allows applications to be easily distributed for installation on mobile devices. Indeed, the .apk file is the file that users are required to download and all the code in an .apk is essentially one application.

### 3.3.1 Features

The features of Android are as explained below:

- Application framework enabling reuse and replacement of components
- Dalvik virtual machine optimized for mobile devices
- Integrated browser based on the open source Webkit engine
- Optimized graphics powered by a custom 2D graphics library; 3D graphics based on the OpenGL ES 1.0 specification (hardware acceleration optional)
- SQLite for structured data storage
- Media support for common audio, video, and still image formats (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF)
- GSM Telephony (hardware dependent)
- Bluetooth, EDGE, 3G, and Wi-Fi (hardware dependent)
- Camera, GPS, compass, and accelerometer (hardware dependent)
- Rich development environment including a device emulator, tools for debugging, memory and performance profiling, and a plug-in for the Eclipse IDE

### 3.3.2 Android Architecture

Figure 3.1 [22] shows major components of the Android operating system. Each section is described in more detail.

#### 3.3.2.1 Applications

It makes the top most layer of android architecture. Built in applications like sms, mms, contacts, browser are part of this layer. Also all the user application code lays in this
These user applications, data and resource files make an android package. One of the main features of Android is that the basic elements of applications can be shared. If permissions are granted for these elements, it is possible to use already developed features of other applications.

This means that Android has been designed to allow application processes to be started when any part of it is needed. In other words, while most systems and applications have a single entry point, or main function, Android applications rely on central components that are instantiated and run when they are needed. There are four major types of Android
components: Activities, Services, Broadcast receivers, and Content providers. Applications are comprised of components. Components communicate using Intent messages. Recipient components assert their desire to receive Intent messages by defining Intent filters. Figure 3.2 displays the communication between Activities using Intent.

![Activity](image)

Figure 3.2. Activity.

There are four types of components used to construct applications:

1. Activities in the system are managed in an activity stack. When a new activity is launched it becomes the top of the stack. Any previous activity will be below it and will not come to the top until the new one exists. Android runs each activity in a separate process each of which hosts a separate virtual machine. The activity can be in one of four states:
   - Active: The activity started, is running and is in the foreground.
   - Paused: The activity is running and visible, but another activity is running on the top. A paused activity maintains state and member information but can be killed by the system in low memory situations.
   - Stopped: It is similar to paused but invisible.
   - Dead: Either the activity is not started or it was in pause or stop state and was terminated by the system to free some memory or by asking the user to do so.

2. Service components provide for background processing when an application’s activity leaves focus and another GUI application comes in the foreground.

3. Broadcast receiver components provide a general mechanism for asynchronous event.

4. Content provider is for sharing of data between applications.

### 3.3.2.2 Application Framework

By providing an open development platform, Android offers developers the ability to build extremely rich and innovative applications. Developers are free to take advantage of the device hardware, access location information, run background services, set alarms, add notifications to the status bar, and much, much more.

Developers have full access to the same framework APIs used by the core applications. The application architecture is designed to simplify the reuse of components; any application can publish its capabilities and any other application may then make use of
those capabilities (subject to security constraints enforced by the framework). This same mechanism allows components to be replaced by the user. Underlying all applications is a set of services and systems, including:

- A rich and extensible set of Views that can be used to build an application, including lists, grids, text boxes, buttons, and even an embeddable web browser.
- Content Providers that enable applications to access data from other applications (such as Contacts), or to share their own data.
- A Resource Manager, providing access to non-code resources such as localized strings, graphics, and layout files.
- A Notification Manager that enables all applications to display custom alerts in the status bar.
- An Activity Manager that manages the lifecycle of applications and provides a common navigation back stack.

3.3.2.3 LIBRARIES

Android includes a set of C/C++ libraries used by various components of the Android system. These capabilities are exposed to developers through the Android application framework. Some of the core libraries are listed below:

- System C library—a BSD-derived implementation of the standard C system library (libc), tuned for embedded Linux-based devices
- Media Libraries—based on Packet Video’s OpenCORE; the libraries support playback and recording of many popular audio and video formats, as well as static image files, including MPEG4, H.264, MP3, AAC, AMR, JPG, and PNG
- Surface Manager—manages access to the display subsystem and seamlessly composites 2D and 3D graphic layers from multiple applications
- LibWebCore—a modern web browser engine which powers both the Android browser and an embeddable web view
- SGL—the underlying 2D graphics engine
- 3D libraries—an implementation based on OpenGL ES 1.0 APIs; the libraries use either hardware 3D acceleration (where available) or the included, highly optimized 3D software rasterizer
- FreeType—bitmap and vector font rendering
- SQLite—a powerful and lightweight relational database engine available to all applications.
3.3.2.4 **Android Runtime**

Every Android application runs in its own process, with its own instance of the Dalvik virtual machine. Dalvik has been written so that a device can run multiple VMs efficiently. The Dalvik VM executes files in the Dalvik Executable (.dex) format. It was designed specifically for Android running in limited environment, where the limited battery, CPU, memory and data storage are the main issues. Android gives an integrated tool “dx”, which converts generated byte code from .jar to .dex file, after this byte code becomes much more efficient to run on the small processors. The conversion from .java to .dex can be visualized in Figure 3.3.

![Figure 3.3. Conversion from .java to .dex file.](image)

3.3.2.5 **Linux Kernel**

This is the bottom layer of android architecture. It is used for memory management, process management. You will never have access to this layer. This provides better performance, better scalability, improved threading support, networking features. The kernel also acts as an abstraction layer between the hardware and the rest of the software stack.

3.3.2.6 **The Manifest File**

Before Android can start an application component, it must learn that the component exists. Therefore, applications declare their components in a manifest file that’s bundled into the Android package, the .apk file that also holds the application’s code, files, and resources.

The manifest is a structured XML file and is always named AndroidManifest.xml for all applications. It does a number of things in addition to declaring the application’s components, such as naming any libraries the application needs to be linked against (besides the default Android library) and identifying any permission the application expects to be granted. But the principal task of the manifest is to inform Android about the application’s components.
3.3.3 Development in Eclipse with ADT

The ADT plugin for Eclipse adds powerful extensions to the Eclipse integrated development environment. It allows you to create and debug Android applications easier and faster. If you use Eclipse, the ADT plugin gives you an incredible boost in developing Android applications:

- It gives you access to other Android development tools from inside the Eclipse IDE. For example, ADT lets you access the many capabilities of the DDMS tool: take screenshots, manage port-forwarding, set breakpoints, and view thread and process information directly from Eclipse.
- It provides a New Project Wizard, which helps you quickly create and set up all of the basic files you’ll need for a new Android application.
- It automates and simplifies the process of building your Android application.
- It provides an Android code editor that helps you write valid XML for your Android manifest and resource files.
- It will even export your project into a signed APK, which can be distributed to users.

3.4. SQLite

SQLite is an in-process library that implements a self-contained, server-less, zero-configuration, transactional SQL database engine. SQLite, like any other SQL Databases, does not have a separate server process. SQLite reads and writes directly to ordinary disk files. A complete SQL database with multiple tables, indices, triggers, and views, is contained in a single disk file. It is not accessed from the client application, but is an integral part of it. It has a small core and is much faster than other databases making it useful for embedded systems. Android uses SQLite as it’s built in embedded database. It is basically used to store local application data.

SQLite is an ACID-compliant embedded relational database management system contained in a relatively small C programming library. The SQLite library can be called dynamically. The application program uses SQLite’s functionality through simple function calls, which reduce latency in database access—function calls within a single process are more efficient than inter-process communication. SQLite stores the entire database (definitions, tables, indices, and the data itself) as a single cross-platform file on a host machine. It implements this simple design by locking the entire database file during writing.

Some of the distinctive features of SQLite are as follows:
- Zero-Configuration: SQLite does not have to be configured or installed; instead, it just works without setup. In case of Android, it is an inbuilt database.

- Serverless: In SQLite, there is not interprocess communication to send requests to the server and to receive results back. With SQLite, the process that wants to access the database reads and writes directly from the database files on disk. There is no intermediary server process.

- Stable Cross-Platform Database File: The SQLite file format is cross-platform. A database file written on one machine can be copied to and used on a different machine with a different architecture.

- Compact: SQLite library is very compact in size which makes it useful for mobile devices thereby making applications compact in size.

- Variable-length records: SQLite uses only the amount of disk space actually needed to store the information in a row. It results in smaller database files thereby making database run faster. The use of variable-length records makes it possible for SQLite to employ manifest typing instead of static typing.
CHAPTER 4

IMPLEMENTATION AND RESULT

Nowadays, Mobile phones have become widely used with excess of over three billion users. Smart phones are getting more popular these days with their touch screen interface as it is easy to use. Currently, mobile devices are used to not only make or receive a call, take photos, and play video games, but also give the special assistance in the business, such as providing internet access, directing access to email and cooperating data, transferring money, and managing bank account. As a result, authentication of mobile devices has become an important concern. At present, mostly PINs and passwords are used for cell phone authentication. Other methods such as graphical patterns are also used in cell phones with Android OS. This technique offers a standard level of protection and provide cheap and quick authentication. Unfortunately, it is not enough to the safeguard mobile device and data access through them because passwords have never been completely protected by the owners; sharing passwords with friends or any other systems are unavoidable problems.

Due to the risks involved in using PINs or passwords, a secured way to protect data is to use a biometric technique. There are many biometric techniques to authenticate a user, but when it comes to cell phones, the technique should be cost effective and space effective (physically as well as in terms of memory). Keystroke Dynamics satisfies both these requirements. Keystroke Dynamics is space effective; since it only requires software and no extra hardware to authenticate users unlike most other biometric techniques. It is also cost effective; the only cost involved for using this biometric technique is the software or a mobile application which can be affordable by common mobile users. This thesis describes the research and implementation of using Keystroke Dynamics for Mobile devices (smart phones). It also describes the advantages and disadvantages of using Keystroke Dynamics.

Mobile Keystroke Dynamics is the research on a biometric technique to uniquely identify a user based on his typing rhythm. The typing rhythm of a user on a mobile device is analyzed to determine unique characteristics in order to distinguish him from other users. As touch screen devices are being used widely, the touch pad of a cell phone is used as a
keyboard instead of a cell phone keypad. Recently many devices only have touch-pad keyboard instead of a regular keypad. The implementation of Keystroke Dynamics for a touch pad keyboard includes a wider range of cell phone devices (smart phones) as opposed to limiting to a fewer devices. This research uses Android Operating System for the implementation of Keystroke Dynamics. Android is one of the leading OS used in smart phone devices these days.

The process of implementing a Keystroke Dynamics on a mobile device is not different from implementing it for a normal computer. It involves taking a samples (user input) from the user, storing the samples (characteristics of the user typing rhythm such as dwell time, flight time and row switch time, error rate) into the database, analyzing the user samples, creating a unique pattern for a user, comparing the actual user login with the samples and then authenticating the user as original or impostor.

A 10 digit cell phone number is used as the user input to be authenticated. In general, cases, an alphanumeric username and a password is used for user input as it provides more factors which helps to uniquely identify a user. Due to limitations in Android OS to provide key-up/ key-down events of an alphanumeric key, only numeric input is used as a method of authentication. Fifteen samples are collected from a user and stored in the database. For each number pressed, key up and key down times are stored. The key up and key down times are processed to calculate the Dwell time (how long a key is pressed) and Flight time (how long it takes to move from one key to another). Row-switch-time is calculated using the flight time. Each time a row is changed while switching keys, the flight time is stored as row switch time. The four rows of the numeric touch-pad are divided into 12 categories depending on the position of the key in a row. E.g. If a row is switched from first row to second row, then it is categorized as row-switch 1; if the switch is from row first to third, then it is categorized as row-switch 2; if the switch is from row first to fourth, then it is categorized as row-switch three and so on until row-switch 12 following all the permutations. For each typo made; a delete key used; the error count is incremented. For each user sample, the data are stored into the database. Once the samples are collected, the data are processed and fed to an algorithm. The algorithm compares the current user input with the mean/standard deviation of the previous samples and checks if it lies within the mean of the
sample values. If it satisfies all the conditions, then the user authentication is successful. If it does not meet the conditions, then the user login is unsuccessful.

A user profile is created to store the user details and validate the user against those credentials. The user is prompted to fill the registration page which creates a login for the user. The registration page includes fields such as First Name, Last Name and Phone number. The phone number can be any phone number not necessarily his own which will be used as a login password for authentication. Figures 4.1 and 4.2 displays the blank Registration screen and example of successful Registration.

Figure 4.1. Registration screen from the Keystroke Dynamics Application on December 15, 2010.

Figure 4.2. Registration screen – success from the Keystroke Dynamics Application on December 15, 2010.

Once the user is registered successfully, the user is asked to input the samples of the login credentials (10 digit mobile number). A total of 15 samples are collected from each user which will be used to create a pattern of user typing. Figures 4.3 shows the screen for entering Sample Data along with an indication saying the number of attempts left.
The samples are validated for correctness of data. If the login data does not match the user details, then the sample is not stored in the database. Only if the sample is valid and has correct data, i.e. the phone number matches with the one stored in the database then the sample is stored in the database. While entering samples, if the user feels that the sample entered was not proper (the time taken to type was more or any typos were made during typing) then the user can delete the previous sample. If the user feels that he wants to clear all the samples, then he can delete all the samples and start entering the samples from scratch. This will enable to record only clean data without any useless data making the authentication more powerful. Figure 4.4 shows the Menu options to clear sample data.

Once the samples are collected, the user is authenticated on his next login attempt. Figures 4.5 displays the Login Screen which shows Registration option for new users.
The login details entered by the user are then compared with the samples using a unique algorithm, which is discussed in detail in a related thesis “Keystroke Dynamics for Mobile Phones – Algorithm and Implementation”. Different factors such as Dwell Time, Flight Time, Row Switch Time, Total time required to login, Number of Typos are combined and compared with the sample data to grant mobile access to the user.
CHAPTER 5

DATABASE SCHEMA

Database structure of keystroke dynamics is fairly simple and optimized. Database used for storing the values is SQLite. Below are the descriptions of all the tables:

- **Registration**: Before submitting the samples, a user is asked for registration. The table stores personal details of each user like First Name, Last Name and Phone Number filled at the time of registration.
  
<table>
<thead>
<tr>
<th>_id</th>
<th>Fname</th>
<th>Lname</th>
<th>Phone</th>
</tr>
</thead>
</table>

- **Login_details**: This table is used to store login details of each user. Each time the user logs in; an entry is made in this table with the timestamp. The entries in this table are not deleted if the user login is unsuccessful. So, the same user can have more than one entry in the table since timestamps will be different.
  
<table>
<thead>
<tr>
<th>_id</th>
<th>Userid</th>
<th>Phone</th>
<th>Timestamp</th>
</tr>
</thead>
</table>

- **Login_time**: The total time required for the user to login is stored in this table. If the user login is unsuccessful, the entry is deleted from this table, thus only successful login attempts of the user are maintained.
  
<table>
<thead>
<tr>
<th>_id</th>
<th>Userid</th>
<th>Total_time</th>
<th>Timestamp</th>
</tr>
</thead>
</table>

- **Dwell_details**: Time elapsed between a key pressed event and key release event of each button is stored in this table with the time stamp and value of the button typed.
  
<table>
<thead>
<tr>
<th>_id</th>
<th>Userid</th>
<th>Key_typed</th>
<th>Dwell_time</th>
<th>Timestamp</th>
</tr>
</thead>
</table>
- **Flight_details**: The table stores the flight details of the keys. The time elapsed between the release of the previous key and pressing of the next key along with the key sequence is stored.

  ```
  _id
  Userid
  Key_num
  Flight_time
  Timestamp
  ```

- **Row_switch_details**: This table basically stores the flight time along with row-switch details. Each time a row is changed when pressing a key (meaning a key in first row is pressed followed by a key in third row), an entry is made consisting of row switch count and flight time aka row switch time. Row switch count is pre-determined; change from first to second row gets ‘1’, first to third gets ‘2’ and so on.

  ```
  _id
  Userid
  Row_switch_count
  Row_Switch_time
  Timestamp
  ```

- **Delete_frequency**: This table stores the count of the button-click event of delete key in each sample or successful login.

  ```
  _id
  Userid
  Key
  Timestamp
  ```
CHAPTER 6

CONCLUSION

In this research, the practical importance of Keystroke Dynamics in mobile devices as a biometric for authenticating users is addressed. Mobile Keystroke Dynamics is the process of analyzing the way users’ type by monitoring keyboard inputs and authenticating based on habitual patterns in their typing rhythm.

To conclude, this research shows that keystroke data collected from the user can be used for authentication. The data is stored as samples and the user can effectively be authenticated to match his typing rhythm using an algorithm. The keystroke data was collected from the user for each key pushed; processed to create factors such as dwell time, flight time, login time, and error rate which are stored in the database. Data stored can easily be used as input to the algorithm, and therefore compares the previous samples with the new sample and identifies the user as authentic or impostor.

6.1 LIMITATION

Implementing Keystroke Dynamics on a touch-pad of a cell phone is different compared to a normal PC keyboard. Reading keystroke events from a touch-pad lacks accuracy due to a latency time and effectiveness of the touch screen feedback. The small touch-pad reduces the factors involved in comparison and increases the chances of typographical errors, leading to inaccurate results.

In addition, Android has a limitation in providing keystrokes for alphanumeric keys. The Android API fails to give back the keystroke events for alphabets and special characters on the touch-pad. This limitation enforces the usage of just a 10-digit phone number as a means of user authentication, rather than an alphanumeric username and a password which proves to be far more effective. The alphanumeric password enables to identify the user in a unique manner, due to additional factors of measurement involved. Hopefully, future versions of Android will provide this feature, in order to help implement the Keystroke
Dynamics on a mobile device, using alphanumeric passwords leading to more accurate results.

6.2 Future Enhancement

The existing application is created using Android which has some limitations to provide keystroke events for alphanumeric characters. Hoping the future versions of Android may provide this feature; this thesis can be developed into a full-fledged application. The addition of alphanumeric passwords will help build a stronger algorithm involving more factors than the current version.

Also, if finger pressure can be captured for each touch, it can add another factor into the algorithm thereby increasing the accuracy. Android was selected as a platform due to its widespread use in the cell phone industry. Other platforms such as iPhone or Windows can be used for development and perhaps provide the missing features from the Android, thus creating a more accurate biometric technique for mobiles.
BIBLIOGRAPHY


