Research Article

Mobile Authentication using Hybrid Modalities (MAHM) in Pervasive Computing

Vasaki Ponnusamy[a], NZ Jhanjhi[b,*], Najm Us Sama[c], Mamoona Humayun[d]

[a]Faculty of Information and Communication Technology, Universiti Tunku Abdul Rahman, 31900 Kampar, Perak, Malaysia;
[b]School of Computer Science and Engineering (SCE), Taylor's University, Malaysia;
[c]Department of Science, Deanship of Common First Year, Jouf University, Sakaka, Saudi Arabia ;
[d] College of Computer and Information Sciences, Jouf university ,Sakaka 72341, Saudi arabia;
*Corresponding Author: NZ Jhanjhi: noorzaman.jhanjhi@taylors.edu.my

Article History: Received: 10 January 2021; Revised: 12 February 2021; Accepted: 27 March 2021; Published online: 28 April 2021

Abstract
Cloud computing facilities using mobile phones has become a necessity more than being an alternative. Cloud storage, social networks and email communications using mobile phones allows for mobility and on the go access to these cloud services. Despite providing such conveniences to the users, cloud access via mobile devices is vulnerable to security breaches especially when a third party gets hold of the mobile phone. A third party may have access to the applications and features of the mobile phone provided that the unauthorized user managed to break the security login of the phone. This gives avenue for the unauthorized user to gain access to the cloud services without further verification. Because most of the cloud services provide a single sign-on features whereby the password is used by cloud services (gmail for example) for the first time only. After the first authentication, an authentication token which is downloaded to the mobile phone is used for subsequent authentications. Although this may sound secure without the use of password, the possibility for an unauthorized user access to the cloud services is still possible because the authentication token is stored in the phone itself and does not require further validations. Moreover, the authentication token is stored as plaintext inside the mobile phone and may be susceptible to security breach. Therefore, an identity management system based on user behavior patterns can be used to further validate the authorized user of the cloud services. The main aim of the research is to design MAHM, for cloud service identity management using human behavior patterns classification. There are various biometrics-based authentication using physiological and psychological have been in place but, environmental and human factors can influence these. In this research, unique behavior patterns will be used to identify distinct characteristics of certain individual. The framework will consist of various user behavior such as keypad tapping behavior, keystroke dynamics and phone swiping behavior as authentication scheme to the cloud applications.

Keywords: Smartphone security, Mobile authentication, Password, Biometric, Machine learning, Keystroke dynamics, Touch dynamics, Tapping, Swiping.

1. Introduction

Cloud computing, as a promising technology, employs the hardware pooling and virtualization to offer the computing and storage services to the public over the public networks like internet. Now, cloud computing is considered a key innovative technology that delivers the computing resources to the customers in a similar way of utility-based services. Most of the internet users utilize the cloud computing to do particular computations and/or store their data remotely over cloud storage devices. On the other side, people nowadays are shifting from the usage of PC's to the mobile devices like smartphones, laptops, and tablets (Aminzadeh et al., 2015). Khali et al., (2014) has stated that, almost 90% people own mobile devices on a global scale and of that almost 50% of them use their mobile devices to access the Internet and web applications. Moreover, many people access the cloud services, like Drop box and Google drive, via the mobile devices (Chiu et a., 2015). However, the ease and convenience of mobile devices has a darker side. This dark side represents the security challenges that accompany the anywhere access of mobile devices. According to Li et a. (2015), security and privacy are vital features that judge the degree of users' loyalty and trust for mobile cloud service. The security challenges include a wide variety of vulnerabilities in the field of access control and identity management, which is the focus of this study. According to La Polla et al. (2013), the security issues in mobile devices like mobile malicious code, malware injections, and credential thefts are increasing significantly. On the other side, the mobile software developers do not consider those security issues through the software development life cycle. Accessing the cloud services via the mobile devices, without considering the aforementioned security issues, is a risky matter for both the users and the cloud providers. One of the major issues in the mobile computing field is the unauthorized access. Evidently, the mobile devices are more vulnerable to the unauthorized access than the other devices (e.g. PC's) due to many reasons (Khali et al., 2014; Khan et al.,2014): i. The ease of capturing or accessing the sensitive data in the lost/stolen mobile devices by a third party, ii. saving the credential and sensitive data (credit cards, passwords, personal identification information in an indecorously secure manner and therefore, these sensitive data become accessible and easy to collect. iii. unauthorized possession of mobile devices is much easier due to theft and misplacement For these reasons, mobile devices are more susceptible to the unauthorized access challenge (Hossain et al. 2015). Thus, the need for a secure identity management system for mobile cloud computing is highly required to protect the mobile devices and the cloud resources from the unauthorized access in the mobility environment

2. Problem Statement

Although there are certain applications available in the market for securing application in the mobile phone and also securing cloud applications, these works have their distinct functions that differ from the motivation of this research. For example, LockApp application available in Apps Store can be used to lock certain mobile application that can only be achieved upon authentication. But this is tedious as users still need to enter the passcode besides the screen lock passwords. Moreover two-factor authentication system available in the market can be used to restrict cloud services to unauthorized users. But this system does not still protect unauthorized user accessing cloud services from the mobile phone as the main intention of this system is only to protect cloud service access from unauthorized locations or devices. Google Authentication Token scheme also protects cloud access from unauthorized users by providing one time password and subsequent access is performed by Authentication Token stored in the mobile phone. This does not block unauthorized users from accessing cloud services via phone because the token is already stored in the phone and anybody who gain access to the phone can directly access the cloud services. This work differs from the above by uniquely identifying the authenticated user of the phone to gain access to cloud services. When the mobile phone is in the possession of an imposter, the proposed system MAHM would be able to differentiate between legitimate user and an imposter. The work mainly

3. Motivation

MAHM system is designed to detect human hand movement on the mobile phone by detecting their hand gesture, movement speed, tapping speed, tapping variations, swiping patterns as each individual has different hand geometric patterns and finger sizes. With that motivation, each user would have unique tapping, swiping and typing patterns and these three are the most common way how a user interacts with the mobile device So this is the main motivation behind this paper that uses a combination of keystroke dynamics and touch dynamins for user authentication. Among the studies (Buschek et al., 2015), it was reported that the use of touch-based features with keystroke dynamic features have significantly improved the accuracy of authentication. In short touch-based features have the discriminating power to differentiate between legitimate and illegitimate users in the context of authentication. Further evidence of the importance of the touch-based features also noted in the recent study in (Krishnamoorthy et al., 2018), where it used a combination of various feature types that include pressure-based features, keystroke dynamics features and device-specific features for authentication in a mobile device to improve the accuracy of authentication. The touch dynamic is suitable for static and continuous authentication and does not require additional hardware. In touch dynamics, the values of different features change a lot with the change of the mobile phone. The proposed MAHM system differs from others in the literature by specifically selecting key important features from keystroke dynamics and tapping dynamics that focuses on cloud authentication. The works in the literature focuses more on first time mobile authentication only to unlock the phone but MAHM focuses on continuous authentication to cloud applications.

4. Related Works

Key stroke dynamics by identifying user PIN (passcode)typing pattern has been investigated by (Killourhy and Maxion, 2009). Research done on the analysis of keystroke dynamics for identifying users as they type on a mobile phone can be found in Bergadano et al ,2002; Clarke et al., 2009; Karatzouni and Clarke, 2007; Nauman and Ali,2010; Zahid et al, 2009). Kestroke dynmics and touch dynamics consists one of the major behavioral features that can be used for authentication purposes. According to Dhage et al., (2015), keystroke dynamics basically looks at the patterns when one typing character on the keyboard and despite being there for long, Dhage et al.(2015) find that there are challenges when it comes to smarphone keystrokes due to the sensors on the smartphones. Keysrtoke is categorized into static whereby a user would have to enter a specific text that is predetermined like a password and dynamic whereby the user is free to type anything the system would continuously monitor the typing behavior of an individual (Sun, Ceker and Upadhyaya, 2017). (Saini, Kaur and Bhatia, 2016). Researchers are looking at many features that can be extracted from keystroke such as key hold time, latency, horizontal digraph, vertical digraph etc (Teh et al., 2016). There are many active research in authentication using keystroke dynamics and mainly focusing using machine learning algorithms such as random forest and Support Vector Machine (SVM) (Zhang, 2015, Corpus et al., 2016, Alshanketi, Traore and Ahmed, 2016, Sun, Ceker and Upadhyaya, 2017, Huang, Hou and Schuckers, 2017, Huang, Hou and Schuckers, 2017, Sharma and Bohra, 2017, Bhatia and Hanmandlu, 2017, Sitova et al., 2016)

In classifying users, there were different methods used that are statistical and machine learning. Among experiments based on a statistical method for authentication, (Dhage et al., 2015) showed 0.81% of EER which is far better than the results achieved in (Deng and Zhong, 2015) but the former uses only 15 users in the experiment. The trend to use machine learning using keystroke dynamics features has been gaining popularity since (Deng and Zhong, 2015). It is evident in the studies (Alshanketi, Traore and Ahmed, 2016; Corpus et al., 2016; Bhatia and Hanmandlu, 2017; Sharma and Bohra, 2017; Sun, Ceker and Upadhyaya, 2017). Among experiments based on this method, the popular classifiers such as RF, NN and SVM were used. At the beginning in the mobile keystroke dynamics study based on this method (Alshanketi, Traore and Ahmed, 2016), DNN was used and achieved 2.3% of EER. Thereafter NN in (Corpus et al., 2016) achieved 61% accuracy and 7% of FAR which desires further improvement. Since then, RF and SVM have been used widely for authentication based on keystroke dynamics features. It has been found that classifier RF in (Bhatia and Hanmandlu, 2017; Sharma and Bohra, 2017) achieved more than 95% of accuracy and performs better than other classifiers in EER, FAR and FRR metrics as well.

Keystroke dynamics is very cost-effective as it does not require additional hardware and the user does not require to put extra effort as it is transparent

In the beginning, the statistical method was used and achieved 10% and below for EER, FAR and FRR in mobile touch and keystroke dynamics study (Tasia et al., 2014),. Thereafter, several studies (Sen and Muralidharan, 2014; Buschek et al., 2015; Krishnamoorthy et al., 2018) were focused on using a machine learning method with a combination of keystroke dynamics such latency feature and touch mobile features such as touch size, touch pressure and touch offset to carry out behavioral authentication. Zhang (Zhang et al., 2015) found that the use of palm and touch combination of gesture yields a good result in the authentication. The authors used a combination sparse representation and dictionary-based classification which is distinct from other research works. Another study by (Sae-Bae et al., 2014) with the sample size of 34 achieved a reasonable EER rate for gesture using palm and fingertips data. The authors focus on the availability of five-finger touch gesture features on Apple devices. The authors utilized the five finger movements and the palm at the center and accuracy above 90% was achieved. Sitova (Sitova et al., 2016) achieved a better EER rate in the walking position as compared to the sitting position. The authors employed scaled Manhattan, scaled Euclidean, SVM and Score Level Fusion for the classification. A good accuracy result was obtained by (Bokor, Antal and Aszl, 2014) using k-NN and SVM classification method whereby the accuracy of 90% was achieved. Based on the observations and performance results, it is noted that touch-based authentication works better with a set of operations instead of relying on one at a time. An error rate close to zero cannot be achieved by just using a single operation. Overall, a touch gesture is very suitable for static and continuous authentication and it has the advantage of not requiring additional hardware. But touch gestures can lead to inconsistent accuracy if user activity is very inconstant. There were other variations of behavior based authentication was performed such as graphical based authentication, progressive authentication etc. (Zheng et al.,Biddle et al., 2010; Chang et al., 2010; Jermyn et al., 1999). De Luca et al., (2012) Sae-Bae et sal., 2012. Riva et al. (2010).

5. Mobile Authentication using Hybrid Modalities (MAHM)

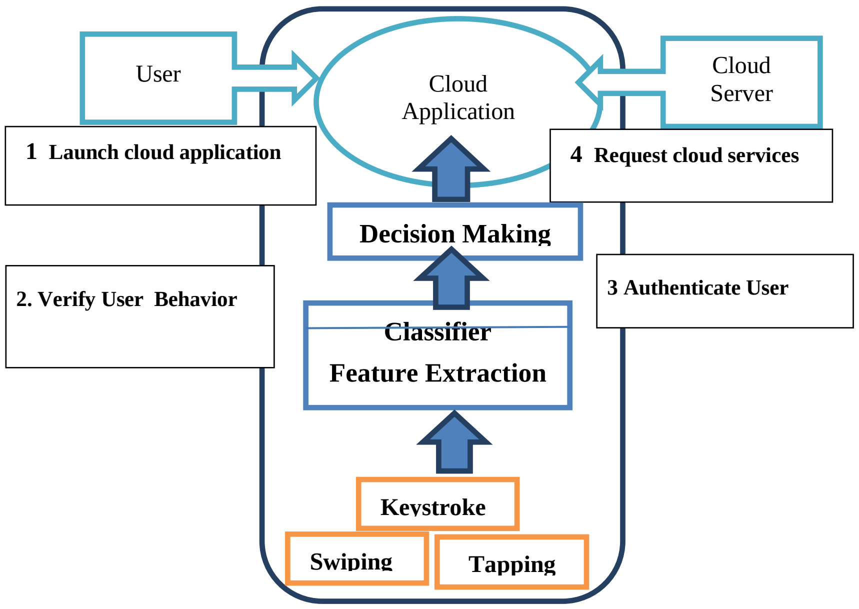
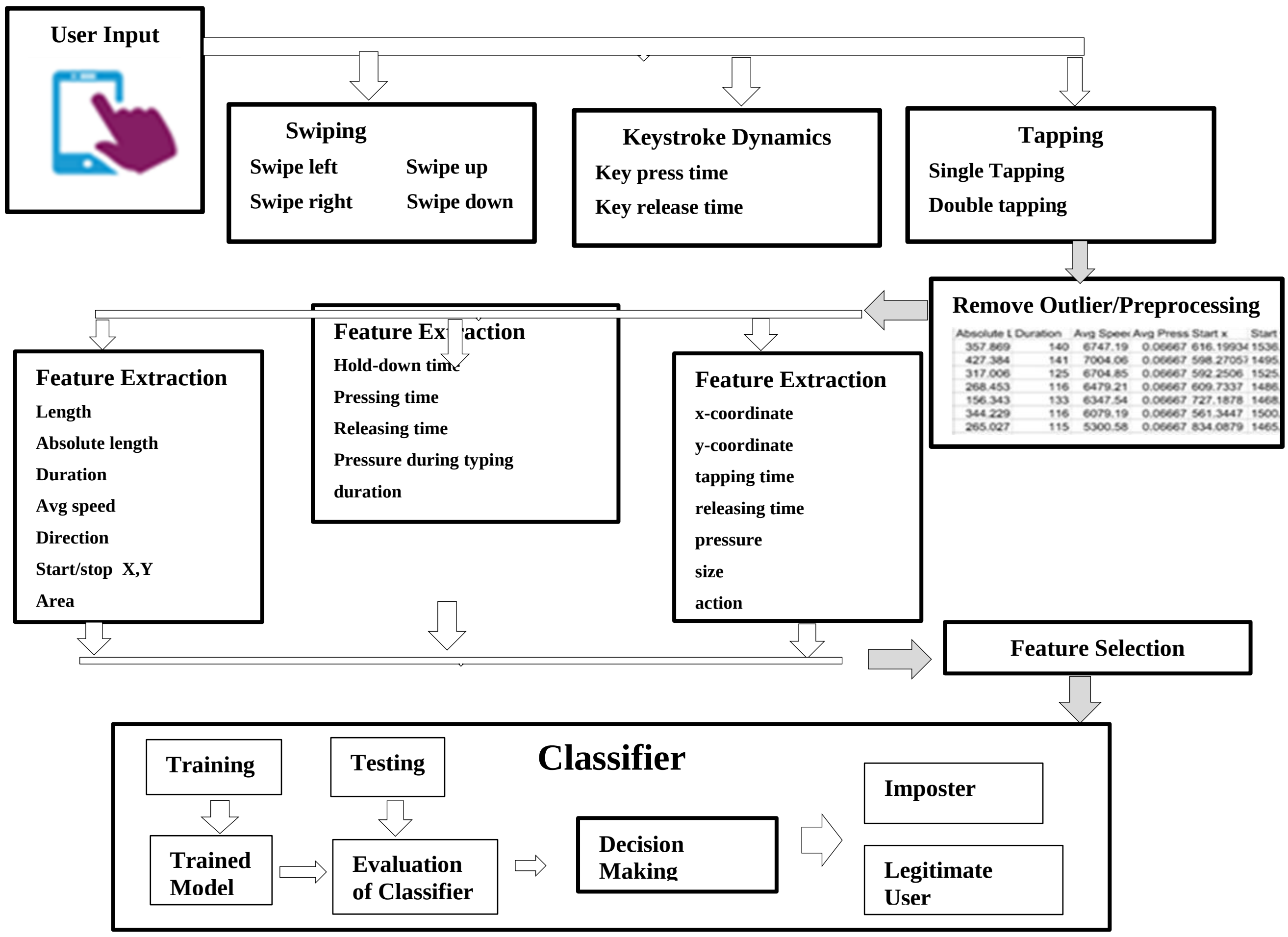


Fig. 1: MAHM Architecture

Cloud applications accessible via mobile phones would not be able to differentiate between legitimate and illegitimate users. Most of the cloud applications request for credentials only at the initial phase of application installation and subsequent authentications is performed by authentication tokens stored in the mobile phone. In that case, any illegitimate users could be using these applications when the mobile phone is in their possession. In order to tackle this issue, MAHM system (Figure 1) through various body of knowledge and literature identified three key behaviour modalities that are commonly used for most of the cloud applications. Keystroke dynamics using keypads or virtual keyboards together with touch dynamics such as tapping and swiping are the most essential and basic features that is applicable for any cloud applications. Whenever a user accesses his/her phone, either tapping, swiping or typing (keystroke) are the most common features used for most of the mobile phone applications and cloud applications. Therefore, this research has identified these three features to create a behaviour based model using machine learning that can uniquely identify certain individuals and can further isolate the unauthorized user from authorized user. In order to do that, an application is created that can extract these features (pressure, time, size and acceleration etc) using the Android API. A classification model is created using machine learning techniques for differentiating the authorized user from the imposter. As can be seen in Figure 2, when a user tries accessing the cloud application, the system would verify the user by using the classification model created and authenticates the user to request for cloud services. MAHM system can also be used for other applications like mobile commerce, mobile learning (accessing cloud storage) and for mobile banking. MAHM is an added security providing continuous authentication on top of traditional two factor authentications that is intended for first time authentication only.



Absolute L	Duration	Avg Speed	Avg Press	Start x	Start
357.869	140	6747.19	0.06667	616.19934	1536
427.384	141	7004.06	0.06667	598.27057	1495
317.006	125	6704.85	0.06667	592.2506	1525
268.453	116	6479.21	0.06667	609.7337	1486
156.343	133	6347.54	0.06667	727.1878	1468
344.229	116	6079.19	0.06667	561.3447	1500
265.027	115	5300.58	0.06667	834.0879	1465

As mentioned in section 5.0, there are three different modalities that is focused in this research to provide a better classification model yielding better accuracy. The three feature extraction, feature selection and classifier systems are discussed separately in the following sections to provide better understandings on the differences of these modalities.

5.1 Feature Selection

Feature selection was considered very important in this research; therefore a step forward feature selection mechanism was adopted. The features were carefully selected by validating the accuracy values with each feature added and those features that were under performing and showing lower accuracy were removed from the classification. Features selection can help in terms of over fitting which could affect the performance. The following are the most suitable features that were identified for all the three different modalities that have been selected for this research.

- Touch Size

Touch size is basically considered the area coverage touched by the user. The area touched varies between different individuals from an adult, child, female and male depending on the fingertip size of the user. The pixel value can be ranged from 0 to 1 add the MotionEven_getsize() function is used in order to extract this in typical android systems. The touch size is usually extracted as the average area, area at the beginning of swipe and area at the end of the swipe. The start area can be wider than the end area as usually in the start the user might swipe wider area and end with lesser touch area.

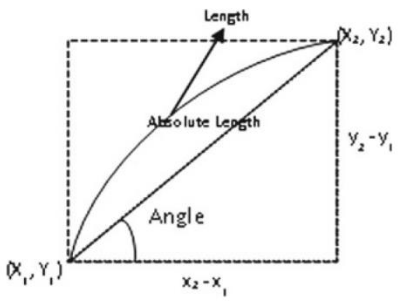
- Touch Pressure

Touch pressure is considered the force applied on the touchscreen and it is directly related to touch area as the more area covered the higher the pressure applied. The touch pressure value ranges from 0 to 1 and function MotionEvent_getpressure() is used to get the pressure value. The touch pressure value is distinctive to every user and it is very hard for imposter to mimic the pressure value of a legitimate user. Average pressure, the pressure at the start of the swipe and pressure at the end of the swipe are extracted and similar to touch size, the touch pressure at the beginning of the swipe could be much stronger than at the end of the swipe due to the area coverage.

- Touch Position (X and Y Coordinate)

The touch position is measured as the X and Y coordinate position of where the user touches the screen. The touch position varies from one individual to another as the touch area, fingertip area and user's palm position differs. Since this is a very discriminative feature, it is very hard to have similar values between a legitimate and illegitimate user. For this simulation, the start X, start Y, end X and end Y values were extracted.

- The Length, Absolute Length and Direction



The absolute length value differs between individuals due to the different hand geometry, so in this case the length and absolute length values were calculated. The length value is extracted from the Android built in function, whereas absolute length is calculated as a straight line from the starting position of the swipe to the end position of the swipe as shown in equation (1). The swipe direction is an angle value of the horizontal line from the swipe starting position and it is calculated as shown in equation (2) and the swipe area is calculated as shown in equation (3).

$$\text{Length} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \qquad (1)$$

$$\text{Angle} = \tan^{-1}\left(\frac{y_2 - y_1}{x_2 - x_1}\right) \qquad (2)$$

$$\text{SwipeArea} = (x_2 - x_1) \times (y_2 - y_1) \qquad (3)$$

- Swipe Duration and Average Speed

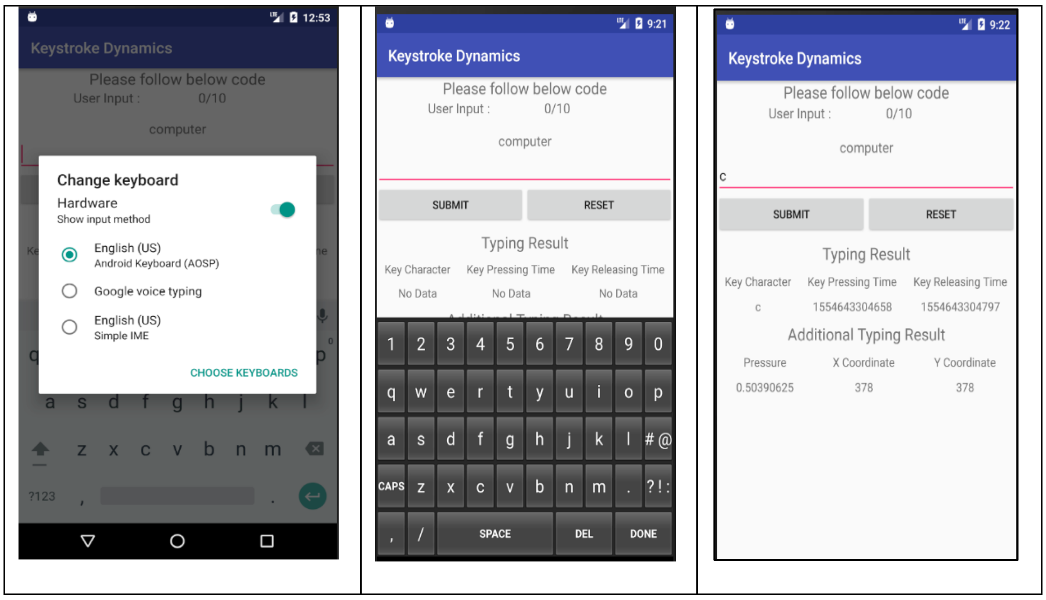
The swipe duration is the total time taken from the start to the end of swipe action. So basically, it is the time difference between the start time and the end time. The getime() function is used from the android system. Function getspeed() is used in order to get the speed value at different instances and it is divided by the number of instances in order to get the average speed value.

- Time

The time value is calculated for the four different instances for the keystroke and the tapping. KeyDown refers to when the user presses the key and KeyUp when user releases the key. Whereas for the tapping Action_Down is user when user performs tapping and Action_Up when user releases the tap. With this associated keystroke and tapping events, the following timing values were obtained as important features: i) key press time, ii) key release time, iii) tapping time, iv) releasing time.

5.2 Keystroke Dynamics

This modality focuses on user typing behaviors to identify their typing patterns for user authentication. The focus area of users for these features would be those performing mobile payments, composing emails and inputting passwords. These are very important credentials or information that needs very reliable medium and therefore when the typing pattern is dissimilar from the authorized user, the system would be able to detect that an imposter is trying to enter the password or making online payments using the legitimate users mobile phone and accounts. There was a separate application developed in order to collect the keystroke data and the application comes with a user-friendly interface. Once the user registers his/her account (Figure 3), the system takes user to the main page with the enrollment, prediction, profile and exit options.

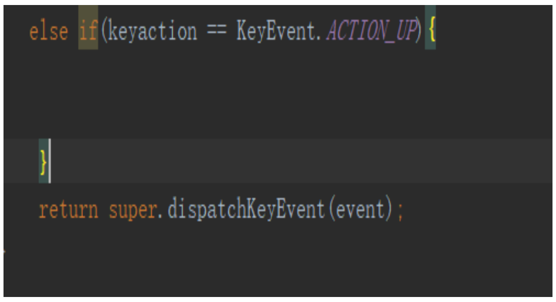


Key Character	Key Pressing Time	Key Releasing Time
No Data	No Data	No Data

Key Character	Key Pressing Time	Key Releasing Time
c	1554643304658	1554643304797

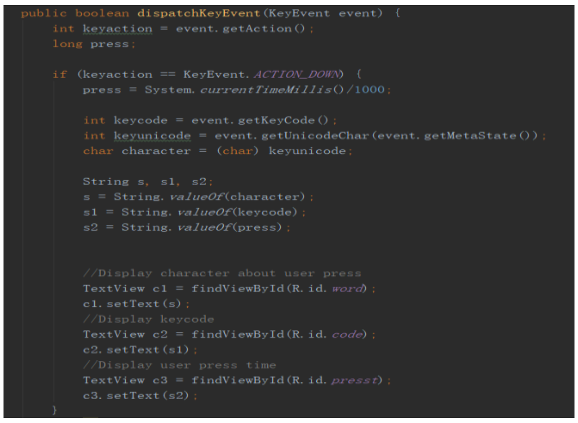
Pressure	X Coordinate	Y Coordinate
0.50390625	378	378

Fig. 3: Keystroke Dynamic Setup



```
else if(keyaction == KeyEvent.ACTION_UP){

}
return super.dispatchKeyEvent(event);
```

Fig. 4(a) Keyboard Dispatch Action



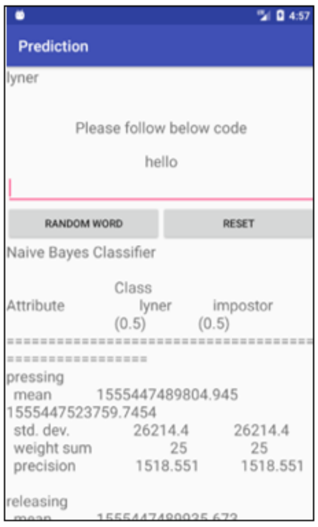
```
public boolean dispatchKeyEvent(KeyEvent event) {
    int keyaction = event.getAction();
    long press;

    if (keyaction == KeyEvent.ACTION_DOWN) {
        press = System.currentTimeMillis()/1000;

        int keycode = event.getKeyCode();
        int keyunicode = event.getUnicodeChar(event.getMetaState());
        char character = (char) keyunicode;

        String s, s1, s2;
        s = String.valueOf(character);
        s1 = String.valueOf(keycode);
        s2 = String.valueOf(press);


        //Display character about user press
        TextView c1 = findViewById(R.id.word);
        c1.setText(s);
        //Display keycode
        TextView c2 = findViewById(R.id.code);
        c2.setText(s1);
        //Display user press time
        TextView c3 = findViewById(R.id.presst);
        c3.setText(s2);
    }
```

Fig. 4 (b) Keyboard Action

In order to input data, the keyboard selections can be made as either default or the custom keyboard. As user enters data using the keyboard, the keystroke dynamic data such as keypress time, keyrelease time, X coordinate, Y coordinate and pressure would be captured as and after certain trials the train button would appear to perform training. The dispatchKeyEvent (Figure 4(a) was used in order to collect the user input where the Key-Down (Figure 4 (b) would detect when the user presses the key while Key-Up (Figure 4 (b) would detect when user release the key. The System.currentTimeMillis () was able to collect user pressing time in order to identify the character typed by the user. In order to predict the user as legitimate or illegitimate, the keystroke data captured (Figure 5) together with input data from the user would be used for testing. Although an imposter could correctly type the password or even other confidential details pretending as legitimate user, the system would still be able to correctly predict the imposter since the keystroke dynamics data captured such as pressing time, releasing time, pressure and duration time varies from the legitimate user as learnt by the system. The profile details would display the user details and the classifier details (Figure 6).

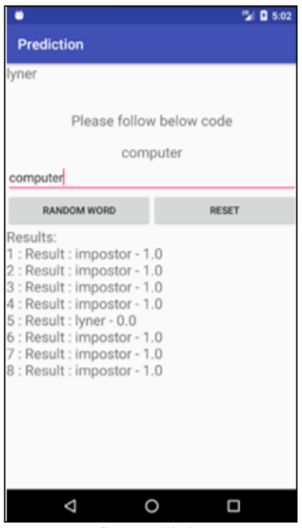
ID	USERNAME	CHARACTER	PRESSING_TIME	RELEASING_TIME	PRESSURE	DURATION_TIME
1	lyner	1	1553801447158	1553801447288	0.50390625	130
2	lyner	u	1553801451623	1553801451740	0.50390625	117
3	lyner	a	1553801455700	1553801455809	0.50390625	109
4	lyner	y	1553801499348	1553801499474	0.50390625	126
5	lyner	q	1553801509191	1553801509287	0.50390625	96
6	lyner	a	1553801514635	1553801514741	0.50390625	106
7	lyner	o	1553801605273	1553801605358	0.50390625	85
8	lyner	4	1553809944703	1553809944845	0.50390625	142
9	lyner	c	1554407008284	1554407008366	0.50390625	82
10	lyner	o	1554407011622	1554407011732	0.50390625	110
11	lyner	n	1554407013871	1554407014050	0.50390625	179
12	lyner	p	1554407029718	1554407029824	0.50390625	106
13	lyner	u	1554407031371	1554407031463	0.50390625	92
14	lyner	t	1554407032790	1554407032905	0.50390625	115
15	lyner	e	1554407035439	1554407035592	0.50390625	153
16	lyner	r	1554407036986	1554407037092	0.50390625	106

Fig. 5 : Keystroke Dynamics Sample Data



```
lyner

Please follow below code

hello

Naive Bayes Classifier

                Class
Attribute           lyner        impostor
                (0.5)         (0.5)
=====================================
=================
pressing
 mean         1555447489804.945
1555447523759.7454
 std. dev.           26214.4         26214.4
 weight sum               25              25
 precision          1518.551        1518.551

releasing
```

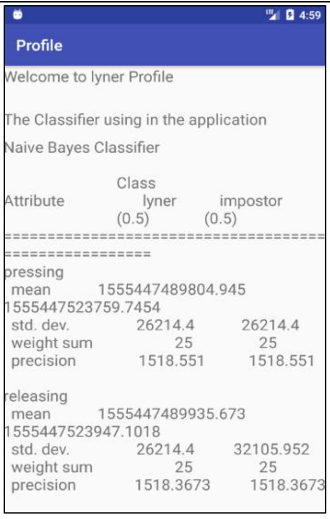
Loading the classifier



```
lyner

Please follow below code

computer

computer

Results:
1 : Result : impostor - 1.0
2 : Result : impostor - 1.0
3 : Result : impostor - 1.0
4 : Result : impostor - 1.0
5 : Result : lyner - 0.0
6 : Result : impostor - 1.0
7 : Result : impostor - 1.0
8 : Result : impostor - 1.0
```

Start prediction

Figure 5. The Result of Predictio



```
Welcome to lyner Profile

The Classifier using in the application

Naive Bayes Classifier

                 Class
Attribute            lyner        impostor
                 (0.5)         (0.5)
=====================================
=================
pressing
 mean         1555447489804.945
1555447523759.7454
 std. dev.           26214.4          26214.4
 weight sum               25               25
 precision          1518.551         1518.551

releasing
 mean         1555447489935.673
1555447523947.1018
 std. dev.           26214.4        32105.952
 weight sum               25               25
 precision         1518.3673        1518.3673
```

Figure 6. Profile Details

Fig. 6: Keystroke Dynamic Results

5.3 Touch Dynamics

Touch dynamics are the second most common action user performs on the mobile phone touchscreen. Most of the time a user performs actions like tap, double-tap, swipe, scroll or pinch to interact with the touchscreen. Touch dynamics is very likely to be used for capturing human behavior due to the distinct features of touch size, touch position, touch pressure, swipe length, swipe duration, swipe size which varies from one individual to another since everyone has different finger size, finger length, finger pressure and the holding methods. Due to these key characteristics, two most common touch dynamics; tapping and swiping has been considered for this research besides the keystroke dynamics.

- Swipe Feature

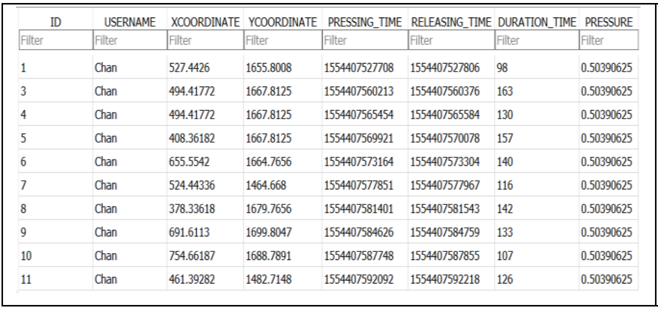
Swipe length, swipe absolute length, swipe starting position, swipe ending position and the time taken for one swipe can be obtained by user swiping up, down, left and right on the mobile touch screen. The author in (Bokor, Antal and Aszl, 2014) developed an application to acquire data for horizontal and vertical strokes. The author used 8 different mobiles and tablets. For classification, the author used K-NN and random forest classifiers using Weka software and achieved an accuracy of more than 95%. From the experiment, the author concluded the most discriminative features i.e. finger area and finger pressure, by measuring in the middle of the swipe, and absolute length between two endpoints.

- Tap and double tap:

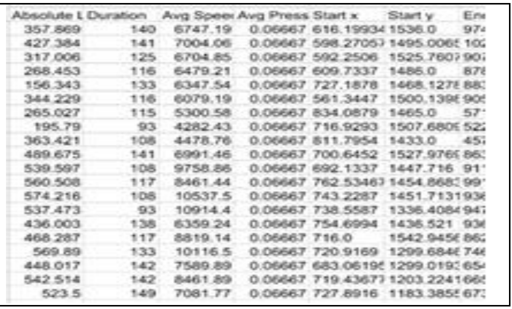
Tap is a gesture that is performed by touching the touch screen to perform a specific function. Tap is performed usually on buttons to open a new something in a window, to submit a form or to send message, etc. Features used for tap are the duration of the tap, touch size, and touch pressure (Sitova et al., 2016). Double-tap is usually used to zoom in or zoom out. From double tap, we can find the same features that can be found with tap and one additional feature velocity between two consecutive taps (Sitova et al., 2016). S. Dhage (Dhage et al., 2015) performed tap features by collecting data using accelerometer, gyroscope, and magnetometer to capture very micro-movements and orientation patterns while performing the tap operation. Grasp resistance (change in movement) and grasp stability (disappearance of finger force) are the two features used for studying the tapping behavior. Based on the implementation results, grasp resistance works better than grasp stability to differentiate the users.

5.3.1 Tapping

Tapping is very crucial for some secured applications like selecting banking details, reading emails or looking for some confidential information on the phone. Therefore, in this setting, tapping details would be collected using a simulated environment whereby the user needs to tap several times for the enrollment phase. The tapping features such as x-coordinate, y-coordinate, tapping time, releasing time, pressure, size and action (Figure 7) would be recorded into the database for training and testing. As shown in Figure 8, the onTouchEvent() method was used in order to recognize the touch events on the mobile screen. This event could identify the tapping and swiping operation on the mobile screen. The ACTION_DOWN could detect the tapping operation and the ACTION_UP could detect tap release opretation. TextView operation was used to obtain the x-coordinate, y-coordinate, size, event time and down time.

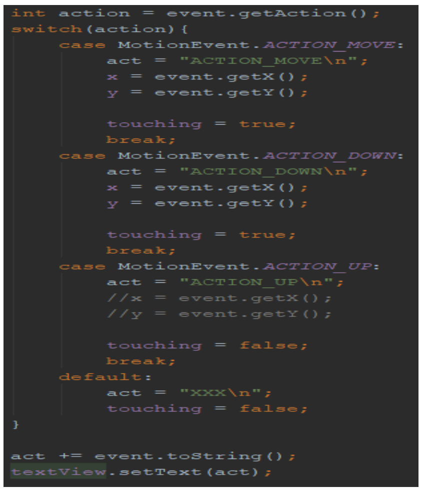


ID	USERNAME	XCOORDINATE	YCOORDINATE	PRESSING_TIME	RELEASING_TIME	DURATION_TIME	PRESSURE
Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	Chan	527.4426	1655.8008	1554407527708	1554407527806	98	0.50390625
3	Chan	494.41772	1667.8125	1554407560213	1554407560376	163	0.50390625
4	Chan	494.41772	1667.8125	1554407565454	1554407565584	130	0.50390625
5	Chan	408.36182	1667.8125	1554407569921	1554407570078	157	0.50390625
6	Chan	655.5542	1664.7656	1554407573164	1554407573304	140	0.50390625
7	Chan	524.44336	1464.668	1554407577851	1554407577967	116	0.50390625
8	Chan	378.33618	1679.7656	1554407581401	1554407581543	142	0.50390625
9	Chan	691.6113	1699.8047	1554407584626	1554407584759	133	0.50390625
10	Chan	754.66187	1688.7891	1554407587748	1554407587855	107	0.50390625
11	Chan	461.39282	1482.7148	1554407592092	1554407592218	126	0.50390625

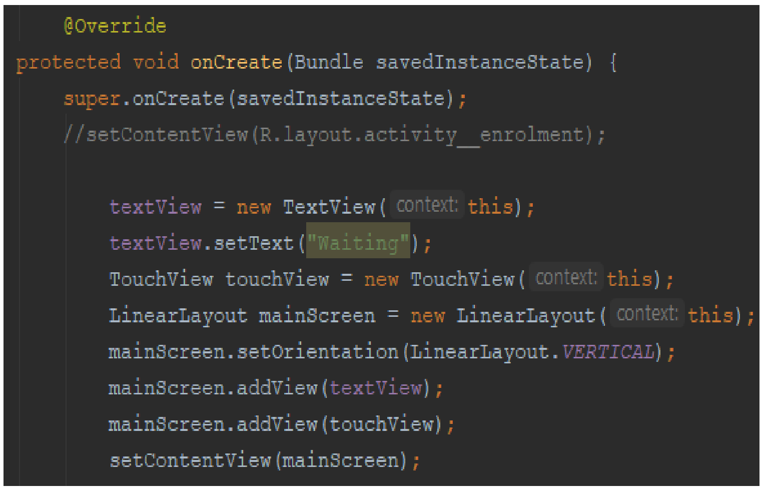


Absolute L	Duration	Avg Speei	Avg Press	Start x	Start y	En
357.869	140	6747.19	0.06667	616.19934	1536.0	974
427.384	141	7004.06	0.06667	598.27057	1495.0065	102
317.006	125	6704.85	0.06667	592.2506	1525.7607	907
268.453	116	6479.21	0.06667	609.7337	1486.0	878
156.343	133	6347.54	0.06667	727.1878	1468.1278	883
344.229	116	6079.19	0.06667	561.3447	1500.1398	905
265.027	115	5300.58	0.06667	834.0879	1465.0	571
195.79	93	4282.43	0.06667	716.9293	1507.6805	522
363.421	108	4478.76	0.06667	811.7954	1433.0	457
489.675	141	6991.46	0.06667	700.6452	1527.9765	863
539.597	108	9758.86	0.06667	692.1337	1447.716	911
560.508	117	8461.44	0.06667	762.53467	1454.8683	991
574.216	108	10537.5	0.06667	743.2287	1451.71319	936
537.473	93	10914.4	0.06667	738.5587	1336.4084	947
436.003	138	6359.24	0.06667	754.6994	1436.521	936
468.287	117	8819.14	0.06667	716.0	1542.9456	862
569.89	133	10116.5	0.06667	720.9169	1299.6846	746
448.017	142	7589.89	0.06667	683.06196	1299.0193	654
542.514	142	8461.89	0.06667	719.43677	1203.22416	665
523.5	149	7081.77	0.06667	727.8916	1183.3855	673

Fig. 7: Touch Dynamics Sample Data



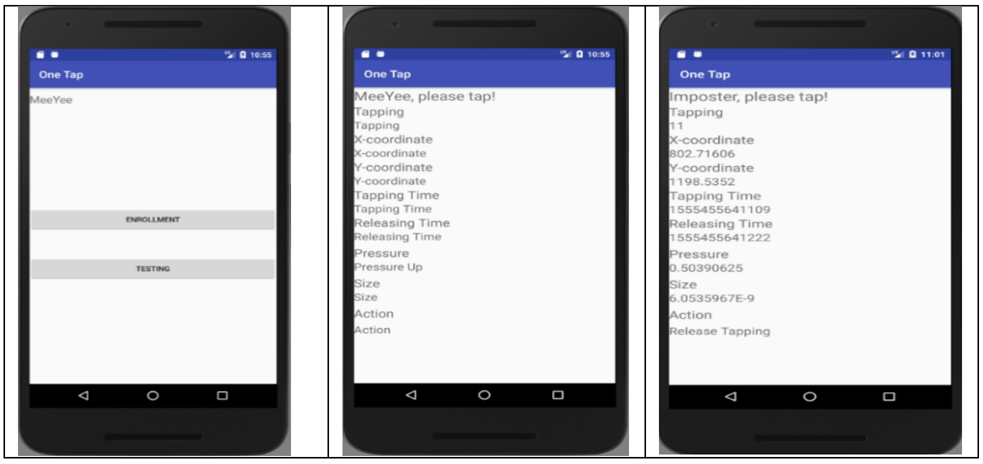
```
int action = event.getAction();
switch(action){
    case MotionEvent.ACTION_MOVE:
        act = "ACTION_MOVE\n";
        x = event.getX();
        y = event.getY();

        touching = true;
        break;
    case MotionEvent.ACTION_DOWN:
        act = "ACTION_DOWN\n";
        x = event.getX();
        y = event.getY();

        touching = true;
        break;
    case MotionEvent.ACTION_UP:
        act = "ACTION_UP\n";
        //x = event.getX();
        //y = event.getY();

        touching = false;
        break;
    default:
        act = "XXX\n";
        touching = false;
}

act += event.toString();
textView.setText(act);
```



```
    @Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    //setContentView(R.layout.activity__enrolment);

        textView = new TextView( context: this);
        textView.setText("Waiting");
        TouchView touchView = new TouchView( context: this);
        LinearLayout mainScreen = new LinearLayout( context: this);
        mainScreen.setOrientation(LinearLayout.VERTICAL);
        mainScreen.addView(textView);
        mainScreen.addView(touchView);
        setContentView(mainScreen);
```

Fig. 8: onTouchEvent() method

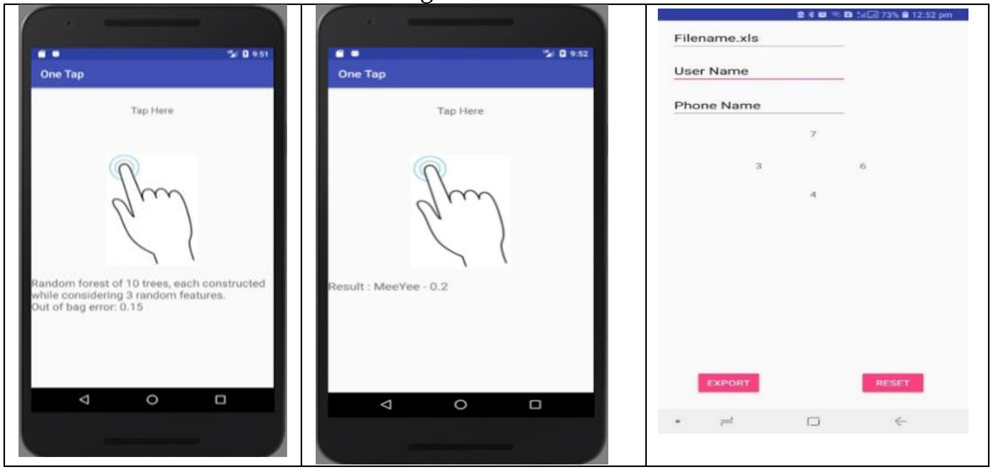
In the prediction phase (Figure 9), verification will be performed whereby the user is asked to tap once and the application will be able to predict whether it is the device owner or an imposter as shown in Figure 10.


One Tap
MeeYee

One Tap
MeeYee, please tap!
Tapping
Tapping
X-coordinate
X-coordinate
Y-coordinate
Y-coordinate
Tapping Time
Tapping Time
Releasing Time
Releasing Time
Pressure
Pressure Up
Size
Size
Action
Action

One Tap
Imposter, please tap!
Tapping
11
X-coordinate
802.71606
Y-coordinate
1198.5352
Tapping Time
1555455641109
Releasing Time
1555455641222
Pressure
0.50390625
Size
6.0535967E-9
Action
Release Tapping

Figure 9: Enrollment Phase


One Tap
Tap Here
Random forest of 10 trees, each constructed while considering 3 random features.
Out of bag error: 0.15

One Tap
Tap Here
Result : MeeYee - 0.2

Filename.xls
User Name
Phone Name

Fig. 10: Touch Operation Obtaining Features

5.3.2 Swiping

In this setting, user can perform swipe left, swipe right, scroll down and scroll up and the swipes would be incremented with every swipe movement. As the example shown in Figure 10, three swipes left, six swipes right, seven scrolls up and four scrolls down are shown and data samples were collected from participants with each participant performing about four hundred swipe movements at all four directions discussed. The data samples were collected during the enrollment process at different time periods to keep the data consistent with different posture holding the phones, with different moods and different positions either standing, lying to sitting etc. Since swipe is a very common gesture used in mobile phones, it was very suitable to collect the swipe features as swipe length, swipe starting position, swipe ending position, time taken from swipe and the swipe absolute length.

6. Results

In order to test the results, 10-fold cross-validation was used in a number of samples for each user was collected. This is a very useful method for small training dataset and it does not fall into overfitting. The datasets ranges from 50 to 400 samples per user. During the testing stage, the accuracy values were obtained with different numbers of training and testing datasets. The training data for each user were extracted from the samples collected for each user and the same applies to the testing datasets. Based on Figure11, 12 and 13 there were different sets of training data used with 50 datasets, 100 datasets and 200 datasets and the testing was done with about 20 % of the total dataset accordingly. Both the training and testing were done with both legitimate (user A) and illegitimate users (user B). Eight different classifiers, Naive Bayes Updatable, Naive Bayes Multinomial, Hoeffding Tree, Random Forest, Decision Stump, Multilayer Perceptron, Naive Bayes and KStar were used for the prediction. As shown in Figure 11, 50 datasets were used for training and and 10 datasets were used for testing which shows the correctly classified instances and also incorrectly classified instances. Naive Bayes Updatable, Naive Bayes Multinomial and Naive Bayes have got 80% correctly classified instances and 20% incorrectly classified instances. However, Hoeffding Tree have got 70% correctly classified instances and 30% incorrectly classified instances while the following classifier Random Forest, Decision Stump, Multilayer Perceptron and KStar have only got 50% of correctly classified instances and 50% incorrectly classified instances. Table 1 shows the prediction results for the legitimate and illegitimate users with each having 50% of values. The results show the instances whereby both users were correctly and incorrectly predicted. It is obvious that Naïve Bayes Multinomial performed better with only one instance of wrong prediction for both users as compared to other classifiers.

Table 1 Prediction with 50 training data

Classifier	Correct Predict userA	Correct Predict userB	Error Predicted userA as userB	Error Predicted userB as userA
Naïve Bayes Updatable	3	5	2	0
Naïve Bayes Multinomial	4	4	1	1
Hoeffding Tree	3	4	2	1
Random Forest	0	5	5	0
Decision Stump	0	5	5	0
Multilayer Perceptron	0	5	5	0
Naïve Bayes	3	5	2	0
KStar	5	0	0	5

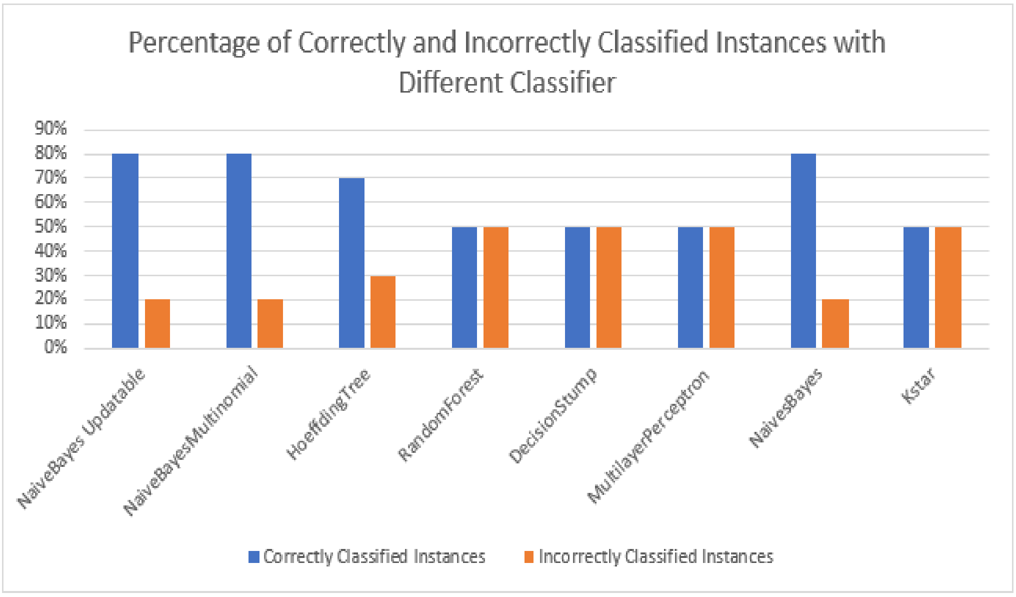


Fig. 11 : Accuracy with 50 training data and 10 supplied test data

As shown in Figure 12, 100 datasets were used for training and and 10 datasets were used for testing which shows the correctly classified instances and also incorrectly classified instances. Hoeffding Tree have got 80% correctly classified instances and 20% incorrectly classified instances. However, Naive Bayes Multinomial have got 70% correctly classified instances and 30% incorrectly classified instances while the following classifier Naïve Bayes Updatable, Random Forest, Decision Stump, Multilayer Perceptron, Naïve Bayes and KStar have only got 50% of correctly classified instances and 50% incorrectly classified instances. Table 2 shows the prediction results for the legitimate and illegitimate users with each having 50% of values. The results shows the instances whereby both users were correctly and incorrectly predicted. There were some changes in the results wiht Hoeffding performing better with only one error predicted for both user A and user B.

Table 2 Prediction with 10 Data with 100 Training Data

Classifier	Correct Predict userA	Correct Predict userB	Error Predicted userA as userB	Error Predicted userB as userA
Naïve Bayes Updatable	5	0	0	5
Naïve Bayes Multinomial	3	4	2	1
Hoeffding Tree	4	4	1	1
Random Forest	0	5	5	0
Decision Stump	0	5	5	0
Multilayer Perceptron	0	5	5	0
Naïve Bayes				
KStar	5	0	0	5

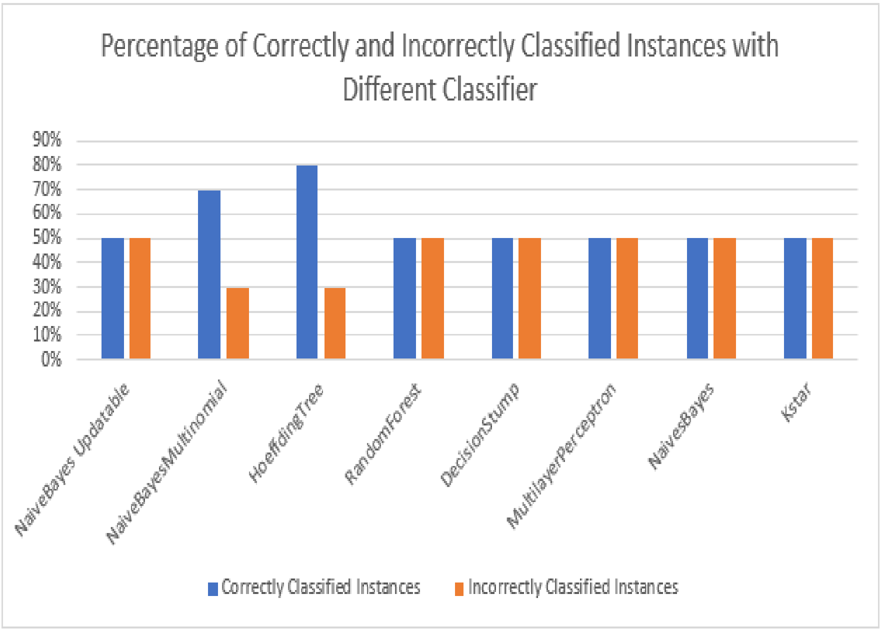


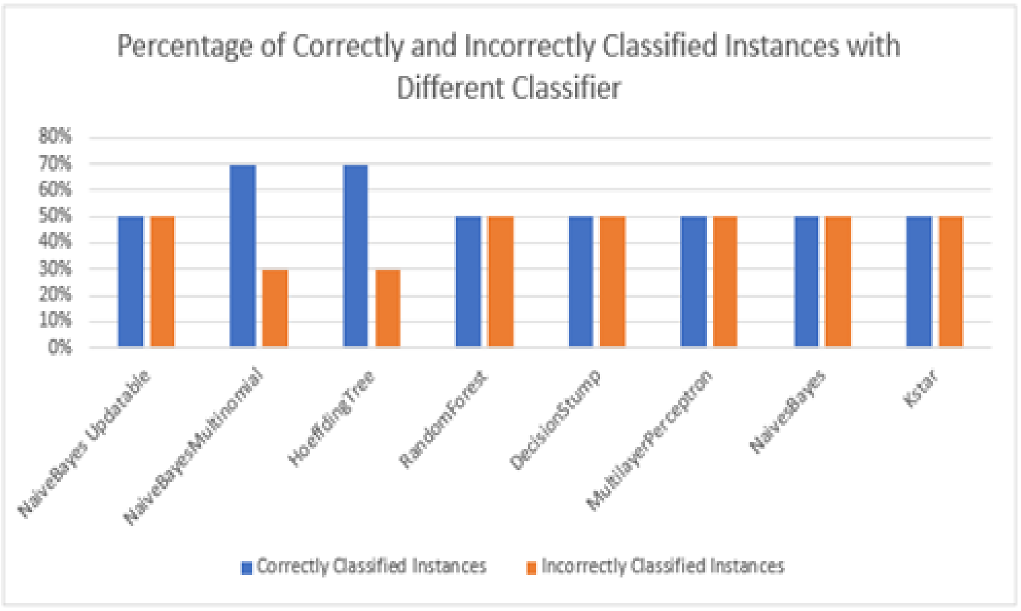
Figure 12 Accuracy with 100 training data and 10 supplied test data

As shown in Figure 13, 200 datasets were used for training and and 10 datasets were used for testing which shows the correctly classified instances and also incorrectly classified instances. Hoeffding Tree and Naive Bayes Multinomial have got 70% correctly classified instances and 30% incorrectly classified instances. The following Naïve Bayes Updatable, Random Forest, Decision Stump, Multilayer Perceptron, Naïve Bayes and KStar have got 50% correct classified instances and 50% incorrectly classified instances. Table 3 shows the prediction results for the legitimate and illegitimate users with each having 50% of values. The results shows the instances whereby both users were correctly and incorrectly predicted. There were some changes in the results with both Hoeffding Tree and Naïve Bayes Multinomial showing two errors predicted for user A and one error predicted for user B.

Table 3 Prediction of 10 Data with 100 Training Data

Classifier	Correct Predict userA	Correct Predict userB	Error Predicted userA as userB	Error Predicted userB as userA
Naïve Bayes Updatable	5	0	0	5
Naïve Bayes Multinomial	3	4	2	1
Hoeffding Tree	3	4	2	1
Random Forest	0	5	5	0
Decision Stump	0	5	5	0
Multilayer Perceptron	0	5	5	0
Naïve Bayes	5	0	0	5
KStar	5	0	0	5

Figure 13 Accuracy with 200 training data and 10 supplied test data



In order to further validate the work, another round of experiments were conducted with two new classifiers, decision tree and IBK. Based on the results in Figure 14, NBM and IBK achieved 100% correct classification whereas NB, DT, DS, RF, MLP, and KStar achieved 50% correctly classified instances and 50% incorrectly classified instances. The prediction results in Table 4 shows that NBM and IBK could correctly classify the user with 0% error rate. So, in this result too NBM seems to be performing better than other classifiers.

Table 4: Prediction of 10 data with 200 training data

Classifier	Correctly Predict User A	Correctly Predict User B	Error Predicted user A as User B	Error Predicted user B as User A
NB	5	0	0	5
NBM	5	5	0	0
DT	0	5	5	0
DS	0	5	5	0
RF	0	5	5	0
MLP	0	5	5	0
IBk	5	5	0	0
KStar	5	0	0	5

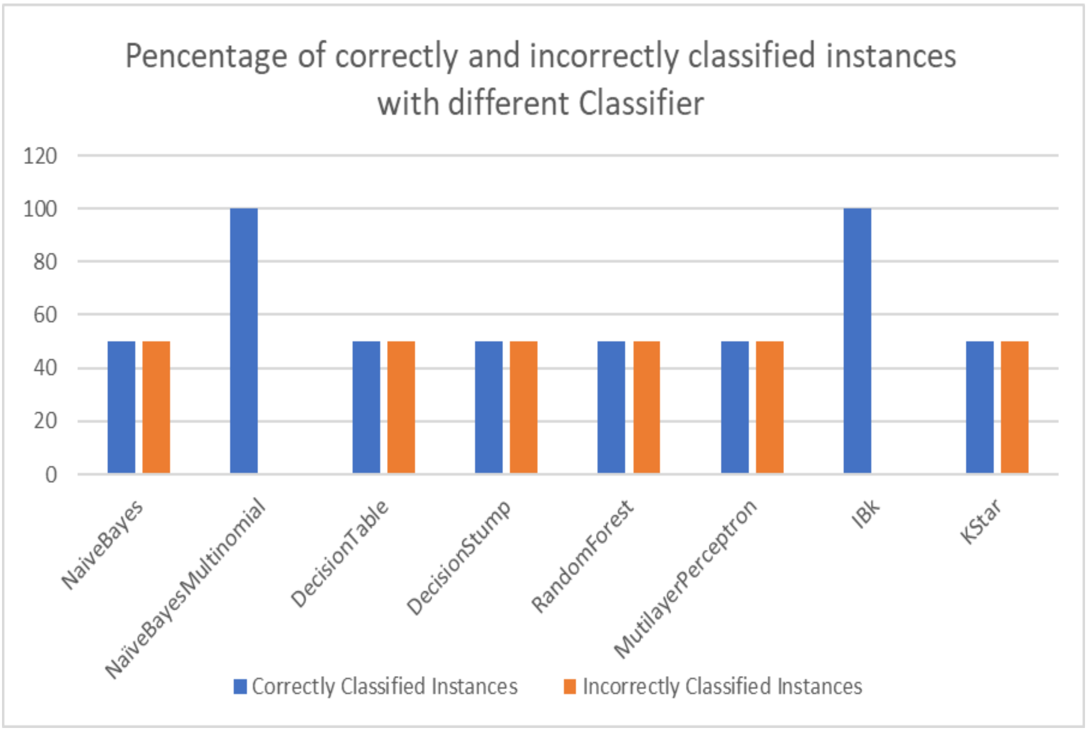


Figure 14 200 Training Data and 10 Supplied Test Data

Table 5

	NB	NBM	DT	DS	RF	MLP	IBk	KStar
10/50	100%	100%	60%	60%	60%	50%	50%	50%
10/100	50%	100%	100%	100%	100%	50%	100%	50%
10/200	50%	100%	50%	50%	50%	50%	100%	50%

Based on the three different tests conducted with different data set values, Naïve Bayes Multinomial and Hoeffding Tree seems to giving constant results but the results can be further improved by having more features for all the three different modalities used, keystroke dynamics, tapping and swiping. It is also very clear that size of the dataset for training can greatly affect the accuracy values achieved. Based on the results NBM seems to be the most constant classifier with producing100% correctly classified instances for all the three different dataset selections and could clearly identify the legitimate and illegitimate user as shown in Table 5.

Summary of accuracy

7. CONCLUSION

This paper proposes a strong authentication protocol for cloud computing that works as a second step verification together with the first-time authentication that is performed in all traditional two factor authentication. Although two factor authentication has been established, as mentioned before, the authentication token stored in the mobile phone would allow any imposter to acquire and use the mobile devices as the authentication will be done automatically without requesting for subsequent credentials. Only first-time authentication requires user credentials after which it is done automatically. So, the proposed MAHM framework, allows continuous authentication by continuously verifying the authenticity of the user by observing the user behavior. The traditional two factor authentications normally only considers the knowledge factor such as password (some you know) and the ownership factor such as access card (something you have) but MAHM considers the inheritance factor (something you are) to be considered the strongest mechanism as it is derived from human characteristics and behaviors that cannot be easily tampered with, manipulated or mimicked by any imposters. Behaviors factors seems to be considered the most secured authentication mechanisms besides other biometrics methods such as fingerprint and retinal scanners. Behaviors authentication is chosen in MAHM so that users are not continuously bothered to enter the credentials like fingerprint or voice but MAHM system can learn the user behaviors by using the built-in sensors embedded in most of the most devises. The sensors would continuously collect the user behaviors using their keystroke, tapping and swiping patterns. MAHM system should be able to uniquely differentiate between the legitimate and illegitimate user by analyzing their behaviors. A combination of three modalities, keystroke, tapping and swiping has been chosen in this research because these three are the most common way how user interacts with the mobile phone. In our previous work, the authentication of each mechanism has been separately analyzed and the results were presented but does not satisfactorily achieve the outcome. Therefore, in this research a combination of these modalities has been chosen and from the body of knowledge, this is first such research that works by looking into a hybrid modality to perform continuous authentication for cloud computing. The results presented shows that NBM could be the best classifier for this kind of hybrid authentication mechanisms and future work would focus more into improving accuracy values by incorporating other modalities.

REFERENCES

1. Alshanketi, F., Traore, I., & Ahmed, A. A. (2016, May). Improving performance and usability in mobile keystroke dynamic biometric authentication. In 2016 IEEE Security and Privacy Workshops (SPW) (pp. 66-73). IEEE.
2. Almusaylim, Z. A., & Jhanjhi, N. Z. (2020). Comprehensive review: Privacy protection of user in location-aware services of mobile cloud computing. Wireless Personal Communications, 111(1), 541-564.
3. Antal, M., Szabó, L. Z., & Bokor, Z. (2014). IDENTITY INFORMATION REVEALED FROM MOBILE TOUCH GESTURES. Studia Universitatis Babes-Bolyai, Informatica, 59.
4. Bergadano, F., Gunetti, D., & Picardi, C. (2002). User authentication through keystroke dynamics. ACM Transactions on Information and System Security (TISSEC), 5(4), 367-397.
5. Bhatia, A., & Hanmandlu, M. (2017). Keystroke Dynamics Based Authentication Using Information Sets. Journal of Modern Physics, 8(9), 1557-1583.
6. Biddle, R., Chiasson, S., & Van Oorschot, P. C. (2012). Graphical passwords: Learning from the first twelve years. ACM Computing Surveys (CSUR), 44(4), 1-41. [17] T.-Y. Chang, C.-J. Tsai, and J.-H. Lin. A graphical-based password keystroke dynamic authentication system for touch screen handheldmobile devices.J. Syst. Softw., 85(5):1157–1165, May201
7. Buschek, D., De Luca, A., & Alt, F. (2015, April). Improving accuracy, applicability and usability of keystroke biometrics on mobile touchscreen devices. In Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (pp. 1393-1402).
8. Clarke, N., Karatzouni, S., & Furnell, S. (2009). Flexible and transparent user authentication for mobile devices. In IFIP International Information Security Conference (pp. 1-12). Springer, Berlin, Heidelberg.
9. Clarke, N. L., & Furnell, S. M. (2007). Authenticating mobile phone users using keystroke analysis. International journal of information security, 6(1), 1-14.
10. Chiu, W. H., Chi, H. R., & Chang, C. Y. (2015). Exploring innovation model of mobile cloud services: A case study of chung-hwa telecom. In Service Systems and Service Management (ICSSSM), 2015 12th International Conference on (pp. 1-5). IEEE.
11. De Luca, A., Hang, A., Brudy, F., Lindner, C., & Hussmann, H. (2012). Touch me once and i know it's you! implicit authentication based on touch screen patterns. In proceedings of the SIGCHI Conference on Human Factors in Computing Systems (pp. 987-996).
12. Dhage, S., Kundra, P., Kanchan, A., & Kap, P. (2015, January). Mobile authentication using keystroke dynamics. In 2015 International Conference on Communication, Information & Computing Technology (ICCICT) (pp. 1-5). IEEE.
13. Hossain, M. A., Bamhdi, A. M. A., & Sanyal, G. (2015). A New Tactic to Maintain Privacy and Safety of Imagery Information. International Journal of Computer Applications, 110(5), 6-12.
14. Huang, J., Hou, D., & Schuckers, S. (2017, February). A practical evaluation of free-text keystroke dynamics. In 2017 IEEE International Conference on Identity, Security and Behavior Analysis

(ISBA) (pp. 1-8). IEEE.
15. Karatzouni, S., & Clarke, N. (2007, May). Keystroke analysis for thumb-based keyboards on mobile devices. In IFIP International Information Security Conference (pp. 253-263). Springer, Boston, MA..
16. Khan, A. R., Othman, M., Madani, S. A., & Khan, S. U. (2014). A survey of mobile cloud computing application models. Communications Surveys & Tutorials, IEEE, 16(1), 393-413
17. Khalil, I., Khreishah, A., & Azeem, M. (2014). Consolidated Identity Management System for secure mobile cloud computing. Computer Networks, 65, 99-110.
18. Killourhy, K. S., & Maxion, R. A. (2009, June). Comparing anomaly-detection algorithms for keystroke dynamics. In 2009 IEEE/IFIP International Conference on Dependable Systems & Networks (pp. 125-134). IEEE.
19. Krishnamoorthy, S., Rueda, L., Saad, S., & Elmiligi, H. (2018, May). Identification of user behavioral biometrics for authentication using keystroke dynamics and machine learning. In Proceedings of the 2018 2nd International Conference on Biometric Engineering and Applications (pp. 50-57).
20. La Polla, M., Martinelli, F., & Sgandurra, D. (2013). A survey on security for mobile devices. Communications Surveys & Tutorials, IEEE, 15(1), 446-471
21. Li, W., Zhao, Y., Lu, S., & Chen, D. (2015). Mechanisms and challenges on mobility-augmented service provisioning for mobile cloud computing. Communications Magazine, IEEE, 53(3), 89-97.
22. Nauman, M., & Ali, T. (2010). Token: Trustable keystroke-based authentication for web-based applications on smartphones. In International Conference on Information Security and Assurance (pp. 286-297). Springer, Berlin, Heidelberg.
23. Nixon, K. W., Chen, Y., Mao, Z. H., & Li, K. (2014). User classification and authentication for mobile device based on gesture recognition. In Network Science and Cybersecurity (pp. 125-135). Springer, New York, NY.
24. Jermyn, I. H., Mayer, A., Monrose, F., Reiter, M. K., & Rubin, A. D. (1999). The design and analysis of graphical passwords. USENIX Association.
25. Riva, O., Qin, C., Strauss, K., & Lymberopoulos, D. (2012). Progressive authentication: deciding when to authenticate on mobile phones. InPresented as part of the 21st USENIX Security Symposium (USENIX Security 12) (pp. 301-316).
26. Sae-Bae, N., Ahmed, K., Isbister, K., & Memon, N. (2012). Biometric-rich gestures: a novel approach to authentication on multi-touch devices. In proceedings of the SIGCHI Conference on Human Factors in Computing Systems (pp. 977-986).
27. Saini, B. S., Kaur, N., & Bhatia, K. S. (2016). Keystroke dynamics for mobile phones: A survey. Indian Journal of Science and Technology, 9(6), 1-8.
28. Sharma, N., & Bohra, B. (2017, February). Enhancing online banking authentication using hybrid cryptographic method. In 2017 3rd International Conference on Computational Intelligence & Communication Technology (CICT) (pp. 1-8). IEEE.
29. Sitová, Z., Šeděnka, J., Yang, Q., Peng, G., Zhou, G., Gasti, P., & Balagani, K. S. (2015). HMOG: New behavioral biometric features for continuous authentication of smartphone users. IEEE Transactions on Information Forensics and Security, 11(5), 877-892.
30. Teh, P. S., Zhang, N., Teoh, A. B. J., & Chen, K. (2016). A survey on touch dynamics authentication in mobile devices. Computers & Security, 59, 210-235.
31. Sen, S., & Muralidharan, K. (2014, January). Putting 'pressure'on mobile authentication. In 2014 seventh International Conference on mobile computing and ubiquitous networking (ICMU) (pp. 56-61). IEEE.
32. Zahid, S., Shahzad, M., Khayam, S. A., & Farooq, M. (2009). Keystroke-based user identification on smart phones. In International Workshop on Recent advances in intrusion detection (pp. 224-243). Springer, Berlin, Heidelberg.
33. Zhang, H., Patel, V. M., Fathy, M., & Chellappa, R. (2015, January). Touch gesture-based active user authentication using dictionaries. In 2015 IEEE Winter Conference on Applications of Computer Vision (pp. 207-214). IEEE..
34. Zaman, N., & Ahmad, M. (2017). Towards the evaluation of authentication protocols for mobile command and control unit in healthcare. Journal of Medical Imaging and Health Informatics, 7(3), 739-742..
35. Zhao, P., Bian, K., Zhao, T., Song, X., Li, X., Ye, F., & Yan, W. (2016). Understanding smartphone sensor and app data for enhancing the security of secret questions. IEEE Transactions on Mobile Computing, 16(2), 552-565.