Mobile User Authentication Based On User Behavioral Pattern (MOUBE)

Hassan Sbeyti *hsbeity@aou.edu.lb*
Faculty of Computer Study
Arab Open University, Omar Bayhoum Street,
Beirut, 2058 4518, Lebanon

Abstract

Smart devices are equipped with multiple authentication techniques and still remain prone to attacks since all of these techniques require explicit user intervention. The purpose of this paper is to capture the user behavior in order to use it as an implicit authentication technique.

In this paper, we introduce a novel authentication model to be used complementary to the existing models; Particularly, the context of the user, the duration of usage of each application and the occurrence time were examined and modeled using the cubic spline function as an authentication technique. A software system composed of two software components has been implemented on Android platform. Preliminary results show a 76% accuracy rate in determining the rightful owner of the device.

Keywords: Security, Implicit Authentication, Behavioral Modeling.

1. INTRODUCTION

The technological advances in all domains are making the use of smart devices in everyday life more imposing. These range from smart phones to laptops, tablets and even i-watches. This field is in continuous development and every newly released generation is opening new possibilities to the engagement with the user's context and increasing security threats. The European Union Agency for Network and Information Security [1] listed in a survey the top ten security information risks for smart phone users. The number one was data leakage resulting from device loss or theft. This result was also featured by the US-CERT (United States Computer Emergency Readiness Team), which also mentioned that the number of new vulnerabilities has jumped 42% from 2009 to 2010.

In order to fight that, smart devices are usually equipped with three authentication factors: something you know, something you have, and something you are. What you know comes as the main security recommendation for any user; that is to set up his phone with a pin or a strong password. But even that level of security can be trespassed if an attacker has enough time and access to the device. From the user's perspective, that type of authentication has a very low usability therefore a user might choose to store his password on the device for easier access and by that compromising its security. Something you have is by proving possession of something external to the system. Common choices for proving possession are: hardware tokens that generate one-time passwords, access to an e-mail address, the mobile device itself can be registered with an application, and then, possession of the device can be used as a something you have authentication factor. Choices for something you know that require a user to carry an additional device are less convenient for the user. One of the reasons for the popularity of mobile device is its convenience. The something you are factor uses biometrics to authenticate users. Biometric based techniques are multiple such as keystroke analysis that was discussed in a research published in the International Journal of Information Security in 2007[2]. This paper identified two typical handset interactions, entering telephone numbers and typing text messages. It was found that neural network classifiers were able to perform classification with average equal

error rates of 12.8%. Based on these results, the paper concludes proposing a flexible and robust framework to permit the continuous and transparent authentication of the user, thereby maximizing security and minimizing user inconvenience, to serve the needs of the insecure and functional mobile handset. Also, in 2009, a paper was published discussing a different form of keystroke dynamics with the finger pressure [3]. This finding has shown that, the finger pressure gives the discriminative information more than keystroke dynamics with the k-NN analytical method. Moreover, using only the finger pressure produces high accuracy of a 99% rate.

Combining multiple biometrics may enhance the performance of the personal authentication system in accuracy and reliability. In Combining fingerprint and voice print biometrics for identity verification: an experimental comparison [4], 13 combination methods were compared in the context of combining the voice print and fingerprint recognition system in two different modes: verification and identification. The experimental results show that Support Vector Machine and the Dempster-Shafer methods are superior to other schemes.

These authentication methods have proven their weakness in terms of usability and also efficiency. These methods are represented in the phones in the form of different screen lock mechanisms. From these mechanisms, we can name a few, such as:

- A simple swipe, which does not provide security at all and is simply used as a screen saver.
- Face unlock where the user provides a shot of his face that is then recognized by the device and used to unlock it. This method has proven its weakness and its incapability of recognizing the user if the surrounding conditions of light mainly do not match the ones on the day he saved the settings.
- Face unlock and voice which combines the facial with the voice recognition. If the user is found in a place where he cannot raise his voice to the same pitch as the one used when he set up this security, then the authentication will fail.
- Pattern which is the most common form of authentication and yet still weak since an adversary can guess the pattern of the user by simply checking the screen of the phone in an appropriate angle to see traces of the finger.
- PIN and password which are considered as a medium to high security is a combination of numbers or characters chosen by the user and required to be entered at every attempt to unlock the screen which can become quite annoying.

The above mentioned methods are becoming more and more annoying for the user since he has to repeat the same action multiple times a day often over 100 times. These types of authentication are user dependent and require his immediate intervention and input in order to proceed. And by that, any explicit action can be memorized by an adversary and used to unlock the device without the owner's consent. Also, once the device is unlocked, the security feature is deactivated even if it was not with its rightful owner.

Therefore, an additional layer of security is required, one that does not require direct user intervention, but works implicitly and continuously to decide whether the user is indeed the authorized one. The proposed system aims at reducing the number of explicit authentication. Its purpose is not to replace the common authentication methods, but rather to complement them. That is, the user can still use his chosen authentication method, but once the phone is unlocked, the implicit authentication takes charge to determine if the user is indeed the owner or an attacker.

In order to be able to decide that, the device has to gather user centric data that will uniquely characterize the owner. As an example of such data is the gestural input. In the paper Biometric-rich gestures: a novel approach to authentication on multi-touch devices [5], a comprehensive set of five-finger touch gestures was defined, based upon classifying movement characteristics of the center of the palm and fingertips, and tested in a user study combining biometric data collection with usability questions. Using pattern recognition techniques, a classifier was built to recognize unique biometric gesture characteristics of an individual. 90% accuracy rate was achieved with

single gestures, and significant improvement was noticed when multiple gestures were performed in sequence. User ratings aligned well with gestural security, in contrast to typical text-based passwords.

Another implicit authentication technique discussed in "Implicit user re authentication for mobile devices" [6] included the observation of user-specific patterns in file system activity and network access to build models of normal behavior. The proposed system was able to distinguish between normal use and attack with an accuracy of approximately 90% every 5 minutes and consumed less than 12% of a typical laptop battery in 24 hours.

The main focus of our study is to extract the behavior to transform it into a biometric signature that can be used to authenticate the user. We will attempt to discover whether it is possible to extract unique user signature from the behavioral pattern to be used as an implicit authentication mechanism. What kind of user centric information (and in what frequency) should be collected in order to detect the user behavioral pattern? How to transform the detected pattern into a unique signature? What correlation methodology should be used to verify the extracted signature?

In this work, we lay foundational work for implicit authentication through the capture of a user's unique behavioral pattern. The proposed system aims at reducing the number of explicit authentication. Its purpose is not to replace the common authentication methods, but rather to complement them. That is, the user can still use his chosen authentication method, but once the phone is unlocked, the implicit authentication takes charge to determine if the user is indeed the owner or an attacker. To achieve this, we introduce a technique by which we capture the signature of the application usage of a user. First, we collect application related data and in particular the duration of use. Next, we use a mathematical algorithm that will convert that data into a function particular to this user. This function will be used at run-time to determine if the user is indeed the rightful owner or an attacker. Our findings support that this is an approach with great potential. Thus, the main contribution of this work is a framework that helps us understand the user behavior and transform it into a unique signature that can be used to authenticate the user. The study provides an insight into quantifying user behavior and using it as a comparison standard. The remaining parts of this report are organized as follows: Chapter II introduces the related work. Chapter III presents in details the different components of MOUBE (Mobile user authentication based on user behavioral pattern), the behavioral pattern extraction and the mathematical model (cubic spline interpolation). Chapter IV presents the experimental results that evaluate the proposed model. Finally, chapter V, gives an overview about future work.

2. RELATED WORK

Implicit authentication is a very broad topic and has been discussed by multiple papers. We will first look into the phone recognition; next we will go through some research concerning the user recognition. These researches are divided between looking into the behavioral pattern of the user, the keystroke analysis, and the gait recognition. This work is an extension of the work started by B. Elhajj ,H. Sbeyti[7][8] (MUSEP) that is based on the same method to generate the user behavior but the MOUBE system differ form the previous work by the following feature:

1. MOUBE is implemented on an android platform; hence it is tested in real condition, where MUSEP uses simulation.
2. MOUBE uses two learning phases to generate a dynamic threshold for every hour for each user, while MUSEP uses a static threshold for all users.
3. Within MOUBE, the decision whether the user is genuine or intruder is based on five previous threshold comparisons, in MUSEP one comparison leads to the final decision.
4. To evaluate accurately the MOBUE system, the number of user tested is three times more than the number used in MUSEP.

2.1 Phone Recognition

When discussing a pattern of usage, the user is the first thing that comes to mind. However, the phone itself can present a pattern of usage that would make it detectable. The paper "Who do

you sync you are? Smartphone Fingerprinting via Application Behavior" [9] tackles that subject in particular. The research looks into the timing and data volume of network traffic generated by a device. They relied on traffic generated by applications such as Facebook, WhatsApp, Skype, Dropbox, and others. For each packet generated by these applications, they recorded the arrival time, the size of the packet, and the direction whether it's an incoming or outgoing packet. Also, they analyzed the burst which represents the peak of data transfers from the same type of connection, for example TCP packets. By using the K-NN classifier, they extracted what they called "fingerprint" of the phone. Following multiple experiments, they concluded that in about 15 minutes, the phone can be recognized with more than a 90% accuracy rate.

2.2 Authentication mechanisms controlled by the phone

Today's mobile devices are equipped with multiple sensors making them prone to attacks. The researches in the past decade have been guided towards improving their security measures and authentication mechanisms. In order to be considered as a "smart" device, Fisher et al. [10] debate in their paper "Smartphones: Not smart enough?" the idea that a phone should be able to scale up or down its authentication mechanisms based on contextual information received from the device sensors. And by that, the phone would be able to assess the risk and match the corresponding authentication mechanism. First, the paper defines high and low risk scenario where the high risk represents the public use of credit card information and the low risk such as saving passwords onto personal devices in order not to enter them at each sign in. Next, they describe four-device context with examples on how the device should behave in low and high-risk scenarios. For example, the device unlock is a common procedure available in all smart phones. After unlocking our device, we have access to all personal information, except those protected by an extra layer of password security. In a high-risk scenario, the context-aware device should at first sense that the user picked up the phone and is moving it towards his face. Then, it should turn on the camera and scan his face for facial recognition to confirm that it is indeed the owner. Next, it should scan for any known Wi-Fi or Bluetooth devices nearby to determine the user's location and assess using the microphone also, if the user is in a crowded space. In a low risk scenario, the phone would just unlock once it recognizes the user's face. Anyone who attempts to unlock to device other than the legitimate user, would have his photo taken and saved within the device. The collection of such data would raise privacy concerns, therefore, the future work will look into minimizing the amount of data collected and aggregate any stored data. Also, they will attempt to understand how mobile device users construct a mental threat model in a variety of contexts and incorporate physical world factors into contextual threat models.

2.3 Behavioral Pattern

User implicit authentication can be achieved by looking into the behavioral pattern of the user. In 2009, in the Palo Alto research center, a paper was published on this same topic. This research [11] introduces the notion of implicit authentication, the ability to authenticate a user to its device based on common actions that the user performs. This paper focuses on the use of this type of authentication for Mobile Internet Devices in particular. Not omitting the fact that implicit authentication can be used in a multitude of other fields such as computers, medical devices to help preserve patients medical records, military equipment and out-of-band transaction verification. This paper evaluates a technique to compute and maintain an authentication score based on recent activities of the user. The scoring varies depending on a set of positive and negative events and depending on the time elapsed. A positive event is defined as a common habit of the user, and when that occurs, the score increases. A negative event is a non-common event for the user, when that occurs, the score decreases. Time elapsing decreases the score if during that time, the user has usually high activity. When the score goes below the event-specific threshold, explicit authentication is needed by the user in order to access that feature of his device. The different data sources that can be used to make authentication decisions are grouped into 3 types: device data, carrier data and third party data. The device data is any data provided by the phone itself such as GPS coordinates, WiFi/Bluetooth connectivity, application usage, biometric-style measurements such as keyboard typing pattern and voice data. The carrier data can be used to know the user's approximate location and phone call patterns. The third party data such as cloud services can also be used since an increasing number of applications are hosted

online. The architecture of the implicit authentication model will be as follows: past behavior will be the key for the learning algorithm, then based on the user model, and recent user behavior, a scoring algorithm will compute a final score based on which it will be decided whether the user is the original device owner. User modeling assumed in this paper is using independent features, where for example, a user's location is independent from its phone call log and any other activity. The data collected to perform this experiment consisted of emails, calls, SMSs, location, contacts, calendar, tasks, memos, alerts, battery level, (un)holstering, USB connections, power on/off, SD card removal/insertions. This data was from a blackberry device, over the period of 3 months. In order to simplify the research, the analysis was done on phone data and location data. Phone data in particular was analyzed based on the lapse of time since previous call, as for location data, they used the interactive clustering algorithm to compute clusters of the most frequently visited locations. The scoring algorithm was applied on this collected data and noticed that the score decreases to zero during the periods knows as active, and during that specific day, were not. Another experiment was conducted where an adversary calls a set of unknown numbers from the user's device, and the score also quickly decreased to zero. As future work, they will attempt to make use of all features for the scoring, and report results on false positive and false negative rates, research methods to model the dependence between different features (i.e., activities) and research methods to model adversarial behavior.

SenSec [12] is an application prototype that constantly collects sensory data from accelerometers, gyroscopes and magnetometers and constructs the gesture model of how a user uses the device. SenSec calculates the sureness that the mobile device is being used by its owner. Based on the sureness score, mobile devices can dynamically request the user to provide active authentication (such as a strong password), or disable certain features of the mobile devices to protect user's privacy and information security. The experiment started with offline user classification by asking a set of 20 random volunteer to repeat 5 to 10 times a certain set of actions, pick up the phone, unlock it, open the email application, lock the phone and return it to the table. The online user authentication consisted of giving a phone for users for 24 hours with the SenSec application running on these phones. A sureness score is calculated. If it falls below a preset threshold while certain operation is performed, an authentication screen will be pop up asking user to enter a passcode. Next these same phones are given to other participants as a negative testing stage. As result, user studies show that SenSec can achieve 75% accuracy in identifying the users and 71.3% accuracy in detecting the non-owners with only 13.1% false alarms. Also, SenSec bears an average 4.96 seconds detection delay.

2.4 Keystroke Analysis

Touch me once and I know it's you! Implicit Authentication based on Touch Screen Patterns [13] paper introduced the idea of a second authentication level. That is, if an attacker has already breached the first level of security, in this case, a lock pattern, the implicit authentication should be able to figure out that the user is an intruder. In order to perform that, the paper suggests to look into the way the user performs the input given the assumption that the intruder already in possession of the user's password pattern. The experiment designed in order to test this idea started by collecting data from 48 users on 4 different locking patterns (horizontal, vertical, vertical with two fingers, diagonal). The data collected was analyzed using dynamic time warping (DTW). This algorithm looks for similarities between sets of data and calculates the cost to match one onto the other. The result is a warp distance that can be used to determine how similar a set is to a reference set. In this work, a sequence consists of a time series of touch screen data (all combinations of X-coordinate(s), Y-coordinate(s), pressure, size, time). The reference set is the one used to identify the owner of the device as a signature of that owner. For each unlock screen, the reference set was created by taking the first 20 unlocks (each one a single unlock) for each user. This first round of testing showed some very low accuracy levels. In the best case, the true negative rate was 57%. This means that a little bit more than four out of ten attacks would have been successful. This was strongly influenced by the time duration of the tests, the environment which was not realistic, and the fact that the participant was informed on how to act with their devices. In the second part of the paper, a more realistic approach was taken for the test. An android application was developed and sent to the participants by email along with a specific

pattern that was assigned randomly. For instance, out of the 26 participants, for whom valid attacks existed, six reached an accuracy of 90% or higher. This second approach increased the overall accuracy by more than 20%. Overall, it can be stated that using touch screen data to identify users works to a certain degree. This is supported by the fact that increasing the threshold for valid authentication attempts improves overall accuracy. As future work, they attempt to improve accuracy of the results, also they will be implementing a prototype based on the presented approach that does the calculation on the mobile device to perform another long-term study based on this application.

Bo, Zhang et Al. feature in their study a framework entitled SilentSense [14]. It consists of tracking the touch actions of the user and combine them with a movement based biometrics in order to verify whether the current user is the owner or guest/attacker. This approach showed that the user can be identified with an accuracy over 99%. For one operation on the device, the framework could capture multiple information, including: the coordinate on the screen of both touch down and release; the duration of one interaction; the sensory data from both accelerometer and gyroscope, the pressure for the finger touching on the screen, and the motion condition of the user. This detection combination was tested in a static and dynamic scenario. In the first, they evaluated the performance through three different applications, including Message, Album, and Twitter. It was noticed that the framework could reach over 80% accuracy within ten event observations, and the owner will be judged within 6 observations. As for the dynamic scenario, the framework collected their processed vertical and horizontal accelerations in the earth coordinate system and combined them with touch event features. After 12 steps, the accuracy to identify a guest can achieve 100% and after 7 steps, the accuracy to identify the owner can achieve 100.

Dividing that kind of data by application seemed to improve accuracy of the results. Looking at the application alone, it contains user centric data more than the phone itself. The application "knows best on when to authenticate and how to authenticate" [15]. In this research, the application developer decides a suitable classifier depending on the type of application. For example, for a browser, a classifier based on touch input behavior would provide more accuracy than one with keystrokes data. This application centric approach achieved over 85\% accuracy rate after 50 training samples.

Classifying movement characteristics of the center of the palm and the fingertips was considered among the promising authentication techniques [16]. The five-finger touch gestures achieved a 90\% accuracy rate in recognizing an owner based on pattern recognition techniques.

Frank, Biedert et Al. propose a framework Touchalytics [17] that relies on touchscreen input as data source. They discussed in their paper the ability to continuously authenticate users based on the way they interact with the touchscreen of a smart phone. That interaction is typically the way the user scrolls text on his phone. It includes sliding horizontally over the screen and sliding vertically over the screen to move screen content up or down. This behavior covers browsing through images or navigating to next screens, or reading emails or documents or browsing menus. Every user interacts differently with his phone in this context and can by that be authenticated according to this particular feature. In order to be able to distinguish between different users, the paper suggests the usage of two different classifiers k-nearest-neighbors (kNN) and a support vector machine with an rbf-kernel (SVM). The kNN classifier takes every new observation (here: a stroke) and locates it in feature space with respect to all training observations. The classifier identifies the k training observations that are closest to the new observation. Then, it selects the label that the majority of the k closest training observations have. SVM generalizes from the observed data, i.e., it forgets the individual observations after training and only saves the decision. Experiments were conducted where a set of users are given a text to read on their phones and their stroke pattern was recorded. Overall, the authentication difficulty seems to increase with increasing temporal distance to the training phase. The individuals in the experiment would complain from having to read a long text and gave up half way. Interestingly, the long-term authentication of the scrolling classifiers is an exception as its median error rate is

lower than for the inter-session authentication. Thereby, depending on the authentication scenario, there is approximately a 0% to 4% chance that the correct user will be rejected or that a false user will be accepted. For some scenarios, this error rate is still too high for the system being directly implemented as is. However, this result demonstrates that touch-based continuous authentication is feasible.

Itus [18] is an open-source framework that can be deployed off-the-shelf and that combines SilentSense and Touchalytics. It provides an application easy to adapt, extensible and with low performance overhead.

2.5 Gait Recognition

Utilizing the physiological and behavioral biometrics along with environmental factors to recognize the owner of a device is one approach in implicit authentication. Assuming that every person has his own movement pattern, that is his manner of walking or moving his feet, then it can be used to authenticate that person. Mobile devices these days are equipped with gait and location sensors that allow them to track this movement pattern. Using correlation to model the data in order to identify the user turned out to be more performing than the FFT (Fast Fourier transform) providing a 7% error ratio with 10% for FFT [19][20]. Also, the paper Pet: when cellular phone learns to recognize its owner [19] used that gait data and applied a different algorithm. Based on the fact that that data is a time series, they chose a variant of Dynamic Time Warping (DTW) algorithm called Fast DTW. The purpose is to assume that the phone will attach to its owner so much that it will be able to distinguish whenever it is being carried by someone other than its owner and take security measure automatically. Their future work included the actual implementation of the recognition system based on this technique.

3. MOUBE

The MOUBE system is composed of two components as depicted in figure1 namely, the learning component and the authentication component. The learning component is composed of two parts: the first part (called learning phase one) is responsible for capturing the user behavior (it runs over one month) and the second phase (it runs over 15 days and its called learning phase two) aims at calculating the user threshold. The authentication component runs in the background infinitely, immediately after executing phase 1 and 2 of the learning component. It acts as implicit authentication system and is activated only when the user finishes using any application.

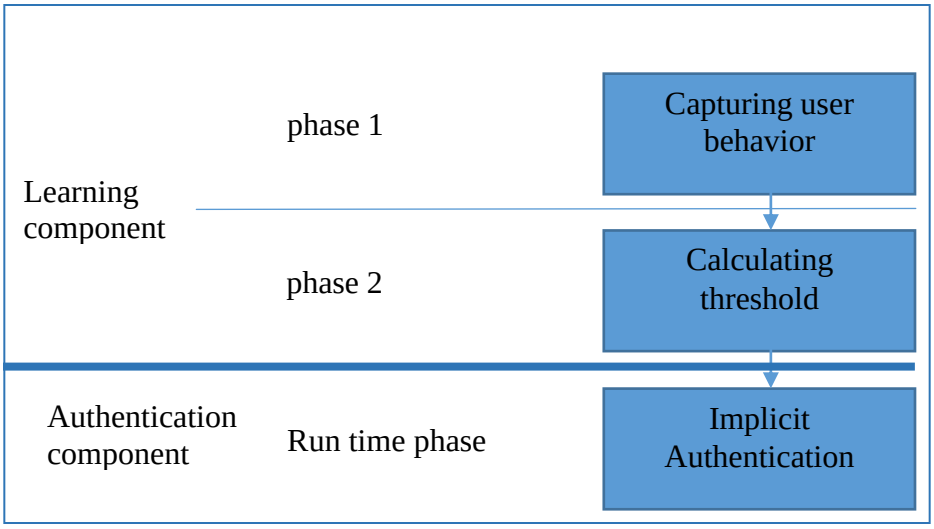

FIGURE 1: MOUBE Architecture.

3.1The Learning Component

The learning component is composed of two parts: the first part aims at capturing the user behavioral pattern and the second component intends to calculate a threshold to be used during the authentication phase for correlation purposes.

User behavior is defined as all kind of user interaction with his/her phone. That is, not limited to, the applications he/she uses, the time he/she uses them, the duration of use, and the order of use.

In this research, we focus our study at analyzing the duration of use of each application. As an example, figure 2 shows the duration of use of the WhatsApp application plotted against the start time of each usage of that application during 5 full weekdays for two users. The data is taken raw and not manipulated in any form. In our study, we assume that each user presents a unique behavior in the duration of use of each application for the same time frame. We will attempt to prove this hypothesis using real collected data and with the support of a mathematical model.

3.1.1 Learning Phase One

Instead of looking at the data at that large scale, we decided to reduce the time scale by looking at individual hours over several days. That is, for example, examining the behavior of a user for one application for one-hour stating at 6:00 PM over a month. Examining the data in this form would create a much more consistent behavior than when looking at it as a whole. Before modeling the captured data, we applied two preprocessing steps:

- Data filtering
- Data smoothing

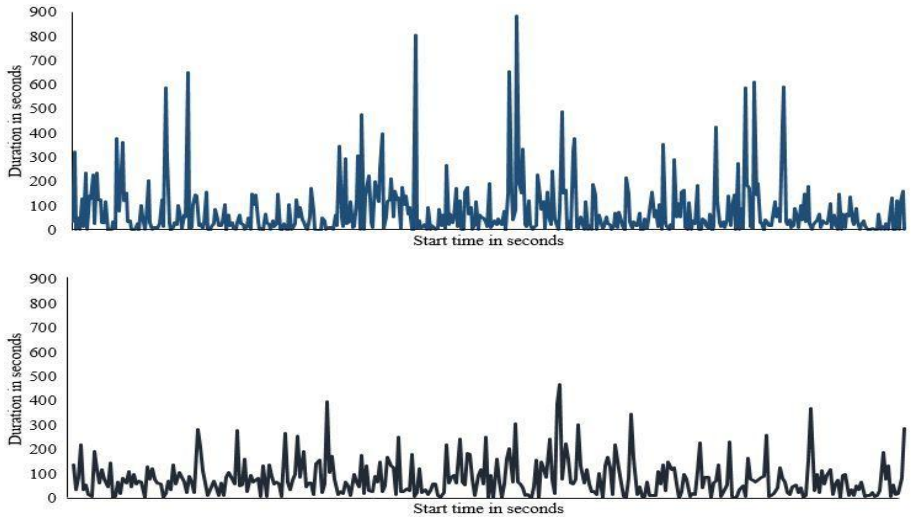

FIGURE 2: WhatsApp duration vs start time for user A and user B.

- Data filtering step

First, the time stamp is divided into hour, minute and seconds. Then a new calculated number is created to convert the minutes and seconds into seconds to retrieve the start time in terms of seconds within that hour (figure 3). Next, the data is filtered by application (WhatsApp in our example as depicted in figure 4).

Seq	Start_Time	Application_Name	Duration	HOUR	MINUTE	SECOND	SEC CONVERTED
1	1/7/2015 0:00	com.whatsapp	22	=HOUR(B2)	=MINUTE(B2)	=SECOND(B2)	=H2*60+I2
2	1/7/2015 0:01	screen off	0	0	1	2	62
3	1/7/2015 0:01	screen off	18	0	1	21	81
4	1/7/2015 0:02	com.whatsapp	28	0	2	46	166
5	1/7/2015 0:03	screen off	2220	0	3	16	196
8	1/7/2015 0:40	screen off	1116	0	40	42	2442
9	1/7/2015 0:59	screen off	3959	0	59	39	3579
12	1/7/2015 2:06	screen off	18362	2	6	41	401
31	1/7/2015 7:12	screen off	1034	7	12	48	768
37	1/7/2015 7:31	screen off	199	7	31	41	1901
43	1/7/2015 7:36	screen off	230	7	36	11	2171

FIGURE 3: Step 1: Time Conversion.

Seq	Start_Time	Application_Name	Duration	HOUR	MINUTE	SECOND	SEC CONVERTED
1	1/7/2015 0:00	com.whatsapp	22	=HOUR(B2)	=MINUTE(B2)	=SECOND(B2)	=H2*60+I2
4	1/7/2015 0:02	com.whatsapp	28	0	2	46	166
62	1/7/2015 7:45	com.whatsapp	19	7	45	10	2710
64	1/7/2015 7:46	com.whatsapp	37	7	46	44	2804
85	1/7/2015 8:01	com.whatsapp	4	8	1	1	61
124	1/7/2015 8:53	com.whatsapp	146	8	53	13	3193
135	1/7/2015 9:00	com.whatsapp	2	9	0	17	17
136	1/7/2015 9:00	com.whatsapp	3	9	0	21	21
162	1/7/2015 9:25	com.whatsapp	44	9	25	46	1546
188	1/7/2015 9:41	com.whatsapp	156	9	41	57	2517
201	1/7/2015 9:48	com.whatsapp	185	9	48	9	2889
212	1/7/2015 9:51	com.whatsapp	88	9	51	48	3108

FIGURE 4: Step 2: Application Filtering.

Next, the duration is filtered to values between 5 and 180 seconds in order to remove the readings that were not meaningful in our approach; this is depicted in figure 5.

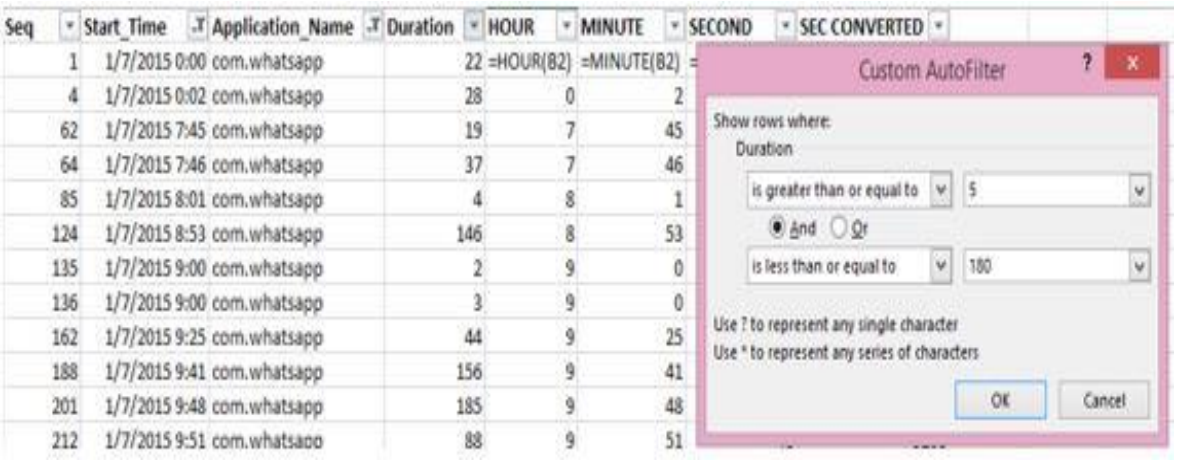

FIGURE 5: Step 3: Filtering for duration between 5 and 180 seconds.

The data is now ready for modelling, as an example; we will take the hour 18:00 (this is depicted in figure 6). The collected data set consists now of the converted start time in seconds, and duration of use in seconds. The columns start time, application name, end time are no longer needed. That data set is ordered in ascending converted start time (first column).

SEC CONVERTED	Duration
48	62
91	20
329	45
543	28
579	27
844	8
1167	6
1200	84
1409	12
1509	47
1512	52
1542	93
1654	68
1702	7
1748	116
1748	34
1953	13
2143	16
2173	68

FIGURE 6: Result of the filtering.

The "sec converted" column can be considered as the abscissa, and the "Duration" its ordinate.

- Data smoothing

To model this data set and avoid fluctuation and negative values in the interpolation, data is sampled at a rate of 8.33×10^{-3} Hz, that is a reading every 2 minutes. Since the user does not necessary use any application at that particular rate, the data is distributed to 31 points by assigning it to the higher start time. The example is shown in figure 7 and in figure 8 The first point is (0, 0). The 2nd start time 64 is less than 120, therefore, the duration 48 is assigned to 120. As for the start time 275, 287, 340, they all fall below 360, so an average of their duration is taken and assigned to 360. As a result: Sometimes, the data acquired does not fill the 31 points that represent an hour. As a solution, midpoints are used to bridge gaps. Using this method, the same sample data used earlier is filtered, and the result is a curve showing one application (WhatsApp), one user, and one hour over 5 consecutive weekdays figure 10. We can notice that this time, the data is less and can be modelled. We will be looking next at a way to quantify that behavior using a mathematical function.

Original start time	Duration of use
64	48
172	59
203	58
275	5
287	5
340	42
414	7
461	39

FIGURE 7: Original Data.

Reading every 2 minutes	Allocated duration
0	0
120	48
240	59
360	17
480	23
600	21
720	40
840	33

FIGURE 8: Original Data Reallocated.

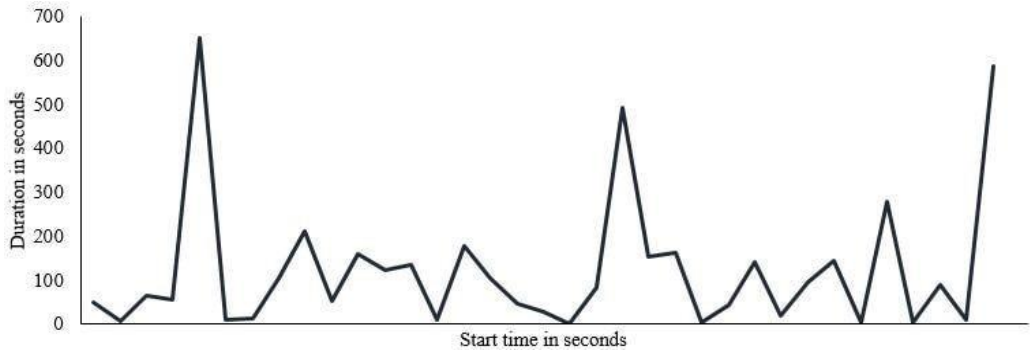

FIGURE 9: WhatsApp 1hour 5 days

3.1.1.1 Cubic Spline Interpolation

The data collected is preprocessed and ready for modelling. Since we have a set of tuples (x , y), a polynomial function is needed. As a first test, on one-time slot, one application was chosen. If this data were modeled using high degree polynomial, the result would be as shown in figure 10, the curve would jump to high results at undesired locations. Also, the curve does not respect the points given to it and is far from being accurate. The plot below was conducted using a 9th degree polynomial.

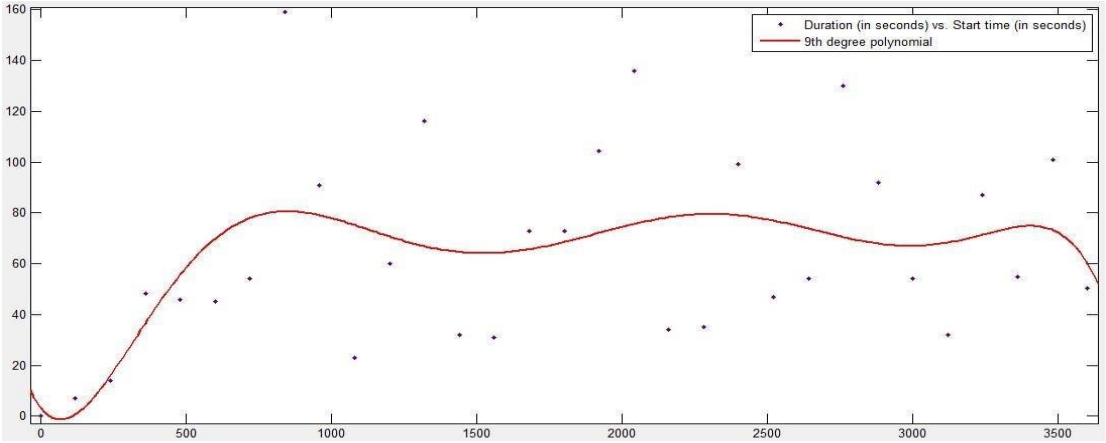

FIGURE 10: Data modelled using 9th degree polynomial.

Given the low accuracy rate with a regular polynomial, we needed a function that would reflect the actual user behavior without compromising its integrity. Using the cubic spline [21][22][23] polynomial leads us to our exact goal by modelling the dataset without an error threshold. The reason for that is that this function is based on individual cubic polynomials that link each 2 points in order to create a smooth curve that passes through all the points. In the plot in figure 11, we can see the same set of points modelled using the cubic spline function.

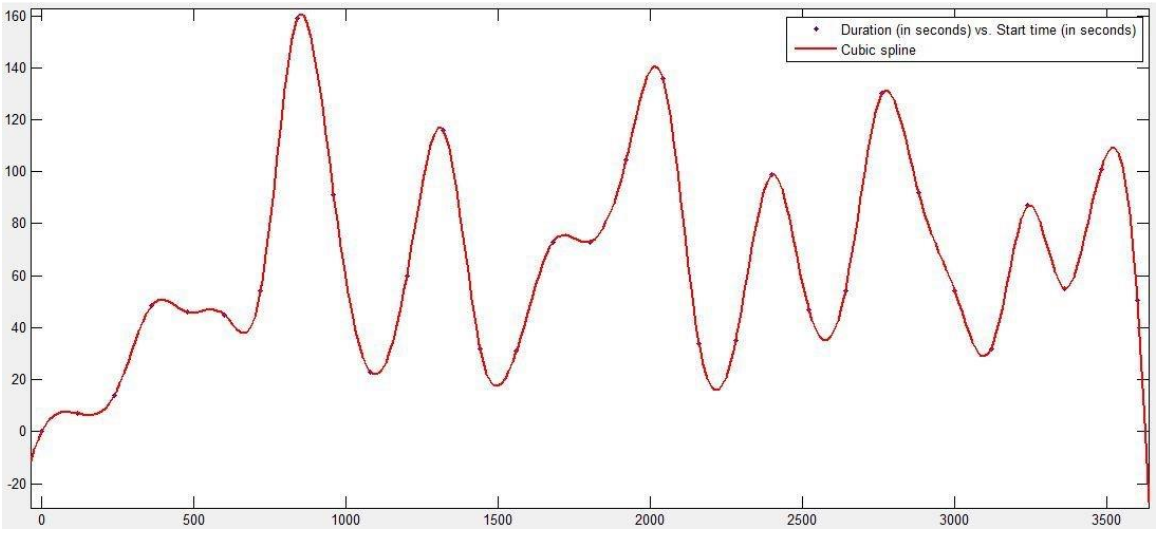

FIGURE 11: Data modelled using cubic spline.

3.1.1.2 Modelling Data Set

The cubic spline interpolation is used to model the dataset as shown in the first graph in figure 12. The graphs show the duration of usage of the same application in seconds, for the same hour, the same dates taken for 4 different users against the 31 points that represent the start time in seconds. From the shape of the function, we can start to notice the de-correlation between users.

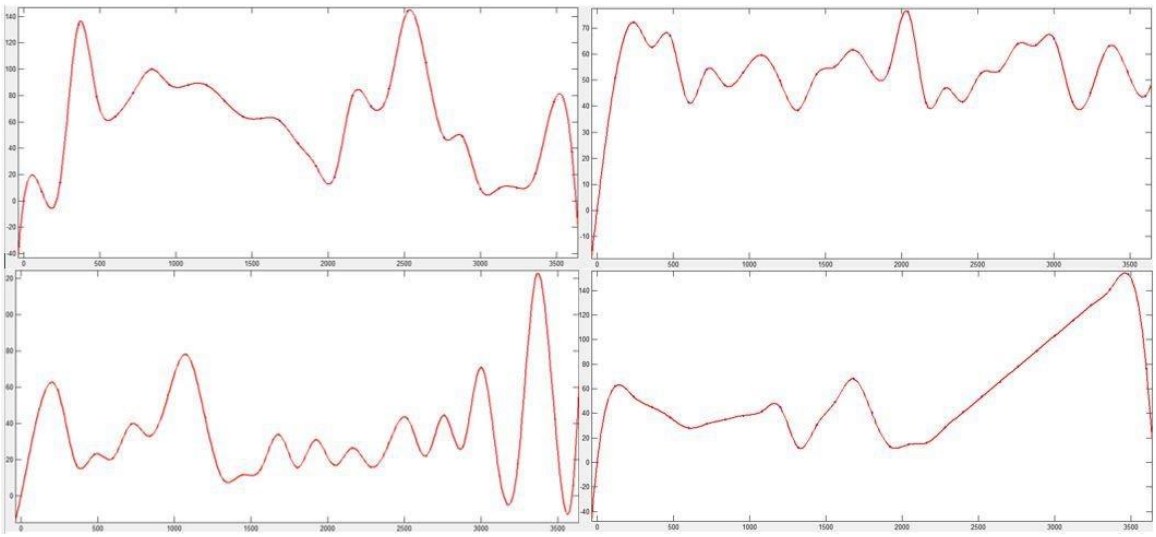

FIGURE 12: Data for four users interpolated using cubic spline polynomial.

This function will be later used to authenticate the user from the duration of use of an application and from the pattern of use that the function has learned throughout this learning phase. At run time, the phone can send to the function the start time of an application and it will return the expected duration of use. Comparing the obtained value with the original run-time value can provide information about the authenticity of the user.

3.1.2 Learning Phase 2 (Threshold calculation)

The learning phase is made up of two phases. During the first phase a cubic spline function (for each user) has been calculated based on data (user behavioral pattern) collected over one month. The aim of the second learning phase is to calculate how much the user behavior fluctuates over an average (called threshold). Below this threshold a user is considered as genuine and above it is considered as an intruder. During this phase, the real captured (x, y) that are collected over fifteen days are clustered by hours as explained in the first learning phase. Each x (of each hour) is feed into the cubic spline function (generated during the first phase) to calculate y'. Then, the absolute value of the difference between the calculated duration (y') and the real duration (y) is recorded. Finally, for every hour, the differences (|y – y'|) are averaged. As depicts figure 13 the average is 24 during the hour 6 is 44 during the hour 7 (the unit of y, y', threshold and x is second). This would give us 24 averages(thresholds) for each user. This is depicted in figure 12 where 24 averages for two users are plotted. Figure 14 depicts also that the average varies form one user to another. The hour averages are considered as thresholds and will be used at run time to authenticate a user. The threshold is considered as a maximum deviation of the user from its normal behavior.

Real Duration	Calculated Duration	Difference	Hour	Minute	Absice	Threshold
Y	Y'	|Y;Y'|			X	AVERAGE
30	34	3	6	1	60	
53	7	45	6	4	240	
8	0	8	6	5	300	
21	16	5	6	6	360	
85	77	8	6	7	420	
12	51	38	6	8	480	
8	0	8	6	19	1140	
14	7	7	6	20	1200	
93	0	93	6	31	1860	
19	0	19	6	34	2040	
62	0	62	6	34	2040	
6	0	6	6	35	2100	
23	13	10	6	47	2820	
23	6	17	6	52	3120	
5	2	2	6	53	3180	
69	126	57	6	54	3240	
23	0	23	7	8	480	24
119	31	88	7	11	660	
65	80	14	7	31	1860	
51	0	51	7	35	2100	
69	15	54	7	36	2160	
93	14	79	7	37	2220	
15	17	2	7	39	2340	
13	70	57	8	1	60	44
25	30	5	8	6	360	

FIGURE 13: Threshold (average) calculation for user 1.

Hour	Threshold User1	Threshold User2
1	0	0
2	0	0
3	0	0
4	0	0
5	0	0
6	46	24
7	0	44
8	9	33
9	43	25
10	25	30
11	23	41
12	36	35
13	8	35
14	31	28
15	33	27
16	33	45
17	23	43
18	44	41
19	36	29
20	26	39
21	38	91
22	52	41
23	30	0
24	0	0

FIGURE 14: Thresholds (averages) for two users over 24 hours.

3.2 The Authentication Component

The authentication component is based on the following algorithm: when new real (x, y) captured (when a user finishes using an application), the start time of the activity (the abscise x) is provided to the cubic spline function (generated during learning phase one) to calculate y' and the absolute value of the difference between the calculated and the real duration (|y – y'|) is compared to the averaged difference (threshold) of the current hour generated during learning phase two. If (|y – y'|) is less than the threshold, a zero (0) is recorded otherwise a one (1) is recoded. Zero means he/she is possibly the owner and one means he/she is possible an adversary. This is depicted in figure 16 in the column "below threshold". The real authentication takes place after five consecutive thresholds comparison (figure 15 & 16). The user is considered as owner if the sum of the ones (1) is less or equal to two (2) otherwise he/she is considered as an adversary. Under the column "Authentication of fire 15 we can notice that there are two results less or equal to two (2), this means that the user is authenticated as the owner of the device.

Real Duration y	Calculated y'	Difference |y>y'|	Hour	Minute	Absice X	Hour Threshold	Below Threshold	Authentication
5	0	5	5	29	1740	0	1	
43	43	0	6	1	60	24	0	
19	16	3	6	3	180	24	0	
26	104	78	6	9	540	24	1	
35	18	17	6	9	540	24	0	2
14	47	32	6	18	1080	24	1	2
67	28	39	6	18	1080	24	1	3
11	124	112	6	38	2280	24	1	4

FIGURE 15: Owner Authentication.

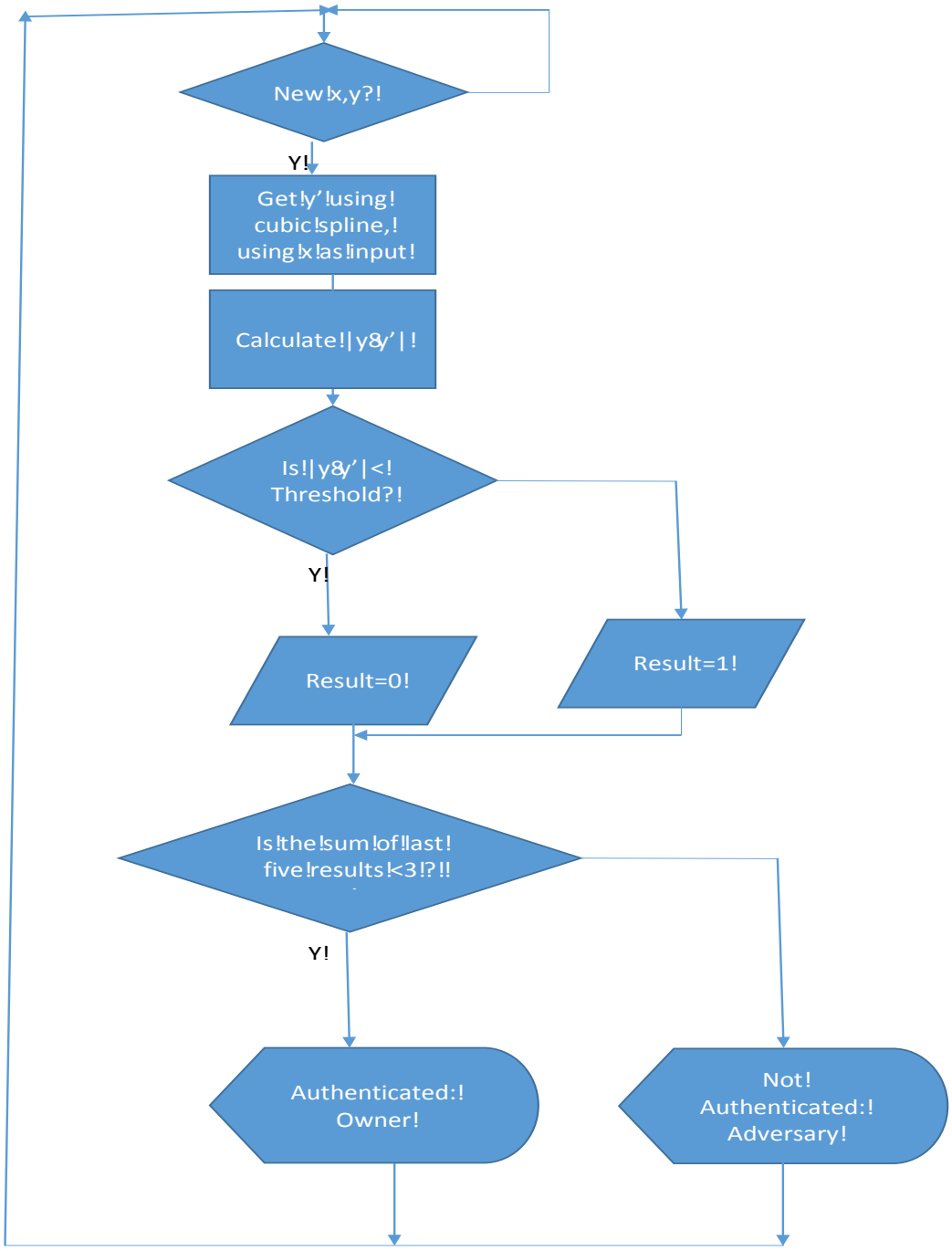

FIGURE 16: Authentication Algorithm.

4. EXPERIMENTAL RESULTS

The MOUBE has been implemented within the middle ware SCAMMP [24] [25] on Android platform. The main goal of SCAMMP is to provide a middleware framework that offers high-level context-aware information through a simple API to the application layer. To implement MOUBE on SCAMMP, only one agent is needed. This agent encapsulates a software sensor namely, the user behavioral sensor. It is defined as a kind of user interaction with his/her phone. The agent records the start time and the duration in seconds.

The MOBE has been evaluated under real time condition; where the application has been installed on the user's mobile phones for a period of two months.

In order to conduct this experiment, we restrict our study to smart phones with android platform. The study consists of collecting user centric data to capture the user behavior. That data is collected from 30 users over a sequence of 60 days. The learning phases run over forty-five (45) days and the authentication period was limited to fifteen (15) days. The users were asked to run the application on their phone and not to stop it till the end of the experiment. No special behavioral requirement was asked of them.

After finishing the learning phases, the authentication phase was started to evaluate the MOUBE algorithm.

First, the following experiments has been executed:

Sixteen (16) users were allowed to run the authentication phase using their cubic spline function and thresholds (that were calculated during the learning phases 1&2) and it is expected to positively authenticate the user as device owner. We except to collected the following values:

- True positive: The user is indeed the owner and
- False negative: The user is the owner of the device but the result suggests he is an adversary.

Second, as for the remaining fourteen (14) users, we switched their cubic spline function and thresholds. The group of these users should not be authenticated by the MOUBE system because they are adversary.

We except to collected the following values:

- True negative: The user is indeed an adversary.
- False positive: The user is an adversary but the result suggests he is the owner.

The results of the above experiment are depicted in figure 17 below. We can notice that we were able to achieve a positive identification of the owner 76%, and the intruder 64% on average. These results are not as high for intruder detection and this is because of the high values of the thresholds for some hours. But considering that this research work traces only one user behavior it has an important achievement. Combining other user behaviors together will for sure provide better identification rates. The MOUBE provides real time decision, immediately after the user finishes using his/her mobile application. While other related works can not produce an immediate decision, even if the identification rates is comparable to some related work, MOUBE opens a new category of real time implicit authentication.

	Average
True positive	76%
False negative	24%
True negative	64%
False negative	36%

FIGURE 17: Results Average.

5. SYSTEM COMPARISON

Providing a real-time implicit user authentication system is not a trivial task, in fact it is a trade-off between accuracy and detection latency. Many implicit user authentication techniques based on behavioral user pattern have been proposed. However, all of them have limitations. Figure 18 depicts the Accuracy in detection the owner, detection latency, and limitations of three different systems. All three systems have comparable accuracy, we do believe that the accuracy can be

further improved by all systems and this is stated in their future works, but the most important thing is the functionality of the system it self. One expects form implicate authentication to act in the background, to be accurate and have real time response. Only the MOUBE system provides a real time system that acts in the background without user intervention, this is depicted in its architecture and in the flowchart of the algorithm. We believe that this is what distinguished MOUBE form the other systems, but the accuracy of detection needs further enhancement. Concerning the limitations of MOUBE, we are further working of making the application available to wide range of users, in order to test the scalability of system accuracy.

System	Behavioral/Soucre	Accuracy	Latency	Limitation
[11]	Device/data,/ carrier/data/and/ third/party/data	Not/avaiabl	Not/a/real/time	False/positive/and/false/ negative/rates/not/avaiable
[12]	accelerometers,/ gyroscopes/and/ magnetometers	75%	Latency/in/fwee/ seconds	The/experiment/was/done/ offline
Ours/ (MOUBE)	Application/ usage	75%	Real/time/respons	Number/of/user/tested/was/ 30/users/only

FIGURE 18: Characteristics of Different Implicit User authentication Schemes.

6. CONCLUSION AND FUTURE

The technological advances in all domains are making the use of smart devices in everyday life more imposing. People are becoming depended from their handheld devices; they almost store every thing we need, contact information, important links, files, e-mails. Securing the access to these devices is becoming a major concern. In order to deal with this concern, smart devices are usually equipped with many authentication techniques, but all of them requires user interventions and hence can be broken. In this research work, we introduce a novel authentication model to be used as complementary to the existing models; Particularly, the context of the user, the duration of usage of each application and the occurrence time were examined and modeled using cubic spline function as authentication technique. A software system composed of two software components has been implemented on Android platform. The system has been installed on thirty (30) users' phones and tested in real situation over a period of two months. Preliminary results show a 76% accuracy rate in determining the rightful owner of the device.

The user behavior can be further expanded to cover things other than the application usage. Everything that is affected by the user can be regarded as user behavior, for instance, the speed of battery drain, the CPU percentage usage, data stream over the Wi-Fi and the mobile data network.

In this research we have considered every application to model a single user behavior, what still can be explored is putting all collected user behaviors in a single matrix. Further, the matrix eigenvalues can be used as a unique signature. One can also use Fourier transformation [26][27] to model the user behavior over long period. In this work, we identify and model the user behavior to be used as implicit authentication, nevertheless, we do believe that there is still a lot to explore in this field.

7. REFERENCES

[1] Enisa.europa.eu.“ Top Ten Smartphone Risks ENISA. Internet: https://www.enisa.europa.eu/activities/Resilience-andCIIP/critical-applications/smartphone-security-1/top-ten-risks, Oct. 25, 2000 [Nov. 24, 2015].

[2] Clarke. “Furnell. Authenticating mobile phone users using keystroke analysis.” International Journal of Information Security, vol. 6.1, pp. 1-14, 2007.

[3] Saevanee, H., and P. Bhattarakosol. “Authenticating user using keystroke dynamics and finger pressure,” in Proc. CCNC Consumer Communications and Networking Conference, 2009.

[4] Wang, Yuan, Yunhong Wang, and Tieniu Tan. “Combining fingerprint and voiceprint biometrics for identity verification: an experimental comparison” in Book Biometric Authentication, Ed. Berlin Heidelberg: Springer, 2004, 663-670.

[5] D.B. Sae-Bae, Napa, et al. “Biometric-rich gestures: a novel approach to authentication on multi-touch devices,” in Proc. SIGCHI Conference on Human Factors in Computing Systems ACM, 2012, pp.

[6] Yazji, Sausan. “Implicit user re-authentication for mobile devices,” in Ubiquitous Intelligence and Computing, Berlin Heidelberg: Springer, 2009, pp. 325-339.

[7] B. El-Hage. “Mobile user signature extraction based on user behavioral pattern.” MSC thesis, Arab Open University - Faculty of Computer Studies, Lebanon, 2015.

[8] H. Sbeyti, B. ElHajj, A. Fadlallah “Mobile user signature extraction based on user behavioral pattern(MUSEP)” International Journal of Pervasive Computing and Communications, Vol. 12, No. 4, October 2016.

[9] Stober, Tim, et al. “Who do you sync you are?: smartphone fingerprinting via application behavior,” in Proc. of the sixth ACM conference on Security and privacy in wireless and mobile networks. ACM, 2013.

[10] Fischer, Ian Timothy, et al. “Short paper: Smartphones: Not smart enough?,” in Proc. of the second ACM workshop on Security and privacy in smartphones and mobile devices. ACM, 2012.

[11] Jakobsson, Shi, et al. “Implicit authentication for mobile devices,” in Proc. of the 4th USENIX conference on Hot topics in . USENIX Association, 2009.

[12] Zhu, Jiang, et al. “Mobile security through passive sensing,” in Proc. of the Computing, Networking and Communications (ICNC). IEEE, 2013.

[13] De Luca, Alexander, et al. “Touch me once and i know it's you!: implicit authentication based on touch screen patterns,” in Proc. of the SIGCHI Conference on Human Factors in Computing Systems. ACM, 2012.

[14] Bo, Cheng, et al. “SilentSense: silent user identification via touch and movement behavioral biometrics,” in Proc. of the 19th annual international conference on Mobile computing and networking. ACM, 2013.

[15] Khan, Hassan, and Urs Hengartner. “Towards application-centric implicit authentication on smartphones,” in Proc. of the 15th Workshop on Mobile Computing Systems and Applications. ACM, 2014.

[16] Sae-Bae, Napa, et al. “Biometric-rich gestures: a novel approach to authentication on multi-touch devices,” in Proc. of the SIGCHI Conference on Human Factors in Computing Systems. ACM, 2012.

[17] M. Frank, R. Biedert, E. Ma, I. Martinovic, and D. Song. “Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication.” IEEE Transactions on Information Forensics and Security, vol. 8(1, pp. 136148, Jan. 2013.

[18] Khan, Hassan, Aaron Atwater, and Urs Hengartner. “Itus: an implicit authentication framework for android,” in Proc. of the 20th annual international conference on Mobile computing and networking. ACM, 2014.

[19] Gafurov, Davrondzhon, Kirsi Helkala, and Torkjel SÃžndrol. “Biometric gait authentication using accelerometer sensor.” Journal of computers, vol. 1.7, pp. 51-59, 2006.

[20] M, Jani, et al. “Identifying users of portable devices from gait pattern with accelerometers,” in Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing. IEEE, 2005.

[21] Monotonic cubic spline interpolation. In Computer Graphics International, 1999. Proceedings (pp. 188-195). IEEE.

[22] McKinley, S. and Levine, M., (1998) Cubic Spline Interpolation. College of the Redwoods.

[23] Klasson, K.T., (2008). Construction of spline functions in spreadsheets to smooth experimental data. Advances in Engineering Software, 39(5), pp.422-429.

[24] H. Sbeyti, M. Malli, A. Fadlallah, K. Tahat, M. Youssef, ” Scalable extensible Middleware framework for context-aware Mobile applications.”, Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA) Vol. 7.3, No. 3, September 2016.

[25] H. Sbeyti et al. “Standardized Scalable Relocatable Context-Aware Middleware for Mobile Applications. Internet: www.scammp-project.info 2014 [June. 28, 2016].

[26] Osgood, B., (2013). Lecture notes for EE 261: the Fourier transform and its applications. Stanford Engineering Everywhere.

[27] Rao, K.R., Kim, D.N. and Hwang, J.J., (2011). Fast Fourier Transform-Algorithms and Applications. Springer Science \& Business Media.