Introducing Touchstroke: Keystroke-based Authentication System for Smartphones

Georgios Kambourakis[1], Dimitrios Damopoulos[2] *, Dimitrios Papamartzivanos[1] and Emmanouil Pavlidakis[1]

[1]Laboratory of Information and Communication Systems Security, Department of Information and Communication Systems Engineering, University of the Aegean, Karlovassi, GR-83200 Samos, Greece.
[2]Department of Computer Science, Stevens Institute of Technology, Hoboken, NJ-07030, USA.

ABSTRACT

Keystroke dynamics is a well-investigated behavioral biometric based on the way and rhythm in which someone interacts with a keyboard or keypad when typing characters. This paper explores the potential of this modality but for touchscreen-equipped smartphones. The main research question posed is whether “touchstroking” can be effective in building the biometric profile of a user, in terms of typing pattern, for future authentication. To reach this goal, we implemented a touchstroke system in the Android platform and executed different scenarios under disparate methodologies to estimate its effectiveness in authenticating the end-user. Apart from typical classification features used in legacy keystroke systems, we introduce two novel ones, namely, *speed* and *distance*. From the experiments, it can be argued that touchstroke dynamics can be quite competitive, at least when compared to similar results obtained from keystroke evaluation studies. As far as we are aware of, this is the first time this newly arisen behavioral trait is put into focus.

KEYWORDS
Keystroke dynamics; Touchscreen; Authentication; Smartphones; Android.

*Correspondence
Dimitrios Damopoulos, Department of Computer Science, Stevens Institute of Technology Hoboken, NJ-07030, USA.
E-mail: ddamop@aegean.gr; ddamopou@stevens.edu

Received . . .

1. INTRODUCTION

Modern mobile devices (smartphones) has rapidly evolved, getting at the same time a lot of attention due to their drastically improved hardware and software capabilities. However, this prevalence has draw the attention of aggressors who, among others, desire to gain access into these devices for stealing or manipulating private information. This growing trend is also backed up by the rise of mobile malware. In fact, malicious software especially designed for smartphones relies upon the lack of strong security mechanisms, thus allowing the exposition of sensitive data, the leak of private information and the exploitation of well-known services. In a recent report [1] it is stated that 33.9% of free of charge third-party iOS applications had some sort of hidden capability to eavesdrop on user's location and 11.2% of them to access personal contacts. DroidDream (for Android) [2] and iSAM (for iOS) [3] are only some characteristic examples of such malicious software that endeavor to steal sensitive data directly from the device. Others, like that in [4], have been designed to compromise popular services like Apple's Siri voice assistant. According to [1], malware attacks have been increased by 155% across all mobile platforms only in 2011.

Additionally, while the number of stolen or lost smartphones has augmented rapidly over the last few years, some of these devices may be used as a stepping stone [5] to spoof the real identity of the attacker. This may be done by taking advantage of the sensitive personal information stored in the device corresponding to its legitimate user. Ireland recently reported that nearly 8,000 smartphones were stolen in the first 6 months of 2012 [6]. Also, only in London, there are around 300 mobile devices stolen every day, while in USA more than 113 mobile phones are lost or stolen every minute [7, 8].

Hence, knowing the increasing risk of mobile threats, designing robust security mechanisms destined to modern mobile devices remains a very demanding task. Generally, there are several important security and privacy parameters which must be taken into careful consideration; user authentication, access control, data integrity, non-repudiation, and content protection are only some of them. Focusing on user authentication, it is undebatable that the current authentication systems offered by major smartphone platforms concentrate solely on point-of-entry mechanism, namely, Personal Identification Number (PIN) authentication. That is, the user is required to enter the correct PIN, usually a number between 4 to 8 digits, before access to the device is granted. As a result, PIN is a security mechanism used only the first time a mobile device is booting into the OS and until the next reboot. Over the years, PINs have evolved to take advantage of the full alpharithmetic keyboards that modern mobile platforms incorporate in an effort to provide complex and secure passcodes. For instance, the latest iOS version supports passphrases of up to 52 characters to be inserted by the user instead of the classic 4-digit PIN. Moreover, with the increasing hardware availability, up-to-date mobile devices are equipped with new sensors such as touchscreens and high-resolution built-in cameras. As discussed in the next section, such advanced hardware has also influenced the mobile OS authentication techniques, which nowadays provide graphical password patterns along with the traditional passcodes.

It is also well-known that PIN authentication systems are facing a number of weaknesses, while remaining vulnerable to brute force and social engineering kind of attacks [9]. Moreover, due to the fact that, at least for the time being, smartphones do not support multiple users, only the one that enters the correct PIN is recognised as the owner of the device, gaining access to all sensitive information stored on it. More importantly, in many cases, mobile users do not adhere to PIN "golden practices", as they never change the PIN, they share it with friends or they write it down on scraps of papers. This makes the PIN-based authentication method inadequate as a protection method for mobile devices [10].

From the above it becomes clear that there is an urgent need for multi-factor intelligent authentication mechanisms, such as those based on keystroke dynamics. It is true that over the last few years researchers have increased their interest in developing intelligent authentication controls relying on biometric technologies for strengthening the security of these devices. To this end, as detailed in section 2, keystroke analysis has been widely used as a fruitful means of profiling (with the intention of authenticating) the legitimate user of a mobile device. In fact, keystroke-powered authentication has been broadly explored in the past but only for mobile devices equipped with physical keyboards neglecting modern smartphones having a touchscreen and a virtual keyboard. Although some recent works in the literature attempted to infer keystroke data from smartphones, none of them actually uses classification features common to legacy keystroke systems for creating typing profiles which can later be used for authenticating the user. Instead, they rely on side-channels (such as motion or sound sensor) to identify the pressed key rather than authenticate the user.

Our contribution: By paraphrasing the term "keystroke" this work introduces touchstroke dynamics (also referred to as "touchstroking" in the rest of this paper). Our basic aim is to explore the potential of this advanced biometric trait in serving as a second verification factor when authenticating the user of a smartphone. In essence, this means that such a system can be used jointly with the login passphrase to make a decision if the person that entered the passphrase is truly the legitimate user of the device. Toward this goal we explore typical scenarios used by the majority of legacy keystroke systems. However, because touchscreen presents a quite different input method (regarding the user behavior when they interact with it) we also consider novel classification features and methodologies along with that employed in typical keystroke analysis. A second major contribution of the work at hand is that the entire experimental procedure has been carried out on a real smartphone in the Android platform. Hence, all results including training and classification times, are directly associated with a typical smartphone hardware and OS. The same applies to performance metrics corresponding to CPU and memory consumption produced by the activation of the proposed solution. In summary, to the best of our knowledge, this work is the first to consider touchstroking and also provide real evidences about its feasibility as a two-factor authentication method in modern smartphones.

The rest of paper is organised as follows. The next section addresses related work. Section 3 details on the authentication scenarios employed, the classification features examined, and the different classification methodologies taken into consideration. Section 4 offers evaluation results, including classification times, and CPU, memory consumption. The last section concludes and provides future directions for this work.

2. RELATED WORK

So far, several research works have directly or indirectly occupied themselves with keystroke dynamics in the realm of mobile devices. In this section we categorise them into three groups: those conducted for devices equipped with a hardware keyboard, touchscreen, and motion sensors.

Studies throughout literature that use keystroke as means to generate better User Interface (UI) models in the context of Human-to-Computer Interaction (HCI) like [11, 12] and others that evaluate keystroke on desktop (fixed) keyboards or computer mouse like those in [13, 14] have been intentionally neglected. The same policy has been followed for works exploiting covert (side) channels, such as electromagnetic and optical emanations, in an attempt to leak out information about which key has been pressed on a keyboard [15, 16]. All the approaches discussed in the following subsections are arranged in chronological and thematic order in Fig. 1.

2.1. Hard keyboard-oriented Keystroke Proposals

In an effort to evaluate the potential to authenticate users by the way they type text messages on a qwerty mobile hardware keyboard, Clarke et al. examined a number of classification algorithms based on Feed-Forward Multi-Layered Perceptron (FF MLPS) perceptron neural networks [17]. Their results have been promising, showing an average classification of 18% Equal Error Rate (EER) and individual users achieving an EER as low as 3.2%. A following work by Clarke and Furnell reports on experimental results conducted on a mobile device interconnected to a laptop [18]. In this experiment, 30 participants were asked to enter data for three scenarios: entry of 11-digit telephone number, 4-digit PINs, text messages. Once more, the classification process has been based on the one developed for the needs of [17], but in this research the authors presented two novel algorithms, namely, Best Case Neural Network and Gradual Training Algorithm, aiming to improve the results. From the study, it is inferred that the second algorithm represents a more plausible technique. On the other hand, the 4-digit and the 11-digit input scenarios achieved an EER of 9 and 8% respectively. Lastly, the 6-digit input scored an EER of 19%. Another research by the same authors explored three input scenarios: entry of a fixed 4-digit number, a fixed 11-digit number, and an alphabetic input [19]. FF MLPs, Radial Basis Function network (RBF) and Generalised Regression Neural Networks (GRNNs) algorithms have been used to classify users in this experiment. It was found that neural network classifiers were able to perform classification with an average EER of 12.8%. Karatzouni and Clarke identified that the hold-time was not a beneficial feature for use on a qwerty mobile device. However, they showed that a combination of both inter-key and hold-time measures would provide better results [20].

The work by Buchoux and Clarke reported on an enhanced keystroke-driven user authentication system. They employed a mobile device not only for capturing typing samples but also to perform the actual authentication [21]. For this purpose, they designed a software able to run on Microsoft Windows Mobile 5. Two types of input password have been proposed; a simple PIN and a strong alphanumeric password. Three classifiers were evaluated: Euclidean distance, Mahalanobis distance and FF MLP. The results suggested that the performance of the classifiers when the password was employed was substantially better than that of PIN due to the increased number of input data.

Saevanee and Bhattarakosol introduced a new metric, namely, finger pressure and combined it with the already existed hold-time and inter-key features to authenticate mobile users [22]. In order to measure the finger pressure the authors used the touchpad of a common netbook acting as a touchscreen. Their study conducted on a sample of 10 participants had an EER of 9% using keystroke dynamics and k-Nearest Neighbors (KNN) classifier.

Zahid et al. collected and analysed keystroke data produced by a sample of 25 mobile device users [23]. The authors proposed a user authentication system that takes into account 6 distinct keystroke features. They demonstrated that these features for different users are diffused and therefore a fussy classifier is well-suited for clustering and classification of those data. The authors argued that their system has a (surprisingly) low average error rate of 2%.

Furthermore, Hwang et al. proposed a Keystroke Dynamics-based Authentication (KDA) system for mobile devices that was able to classify users based on a 4-digit PIN [24]. They correctly highlighted that 4-digit number cannot provide sufficient data for a reliable authentication system. So, they came up with the idea that one way to cope with the lack of data quantity is to improve data quality. For this reason, the authors adopted an input method supported by “artificial rhythms” and “tempo cues”. They experimented with a standard keypad mobile device and found that the proposed strategy reduces EER from 13% to 4%.

The work by Campisi et al. focused on keystroke dynamics within the context of secure user authentication using a numeric mobile hardware keyboard [25]. They employed a statistical methodology able to produce satisfactory verification rates of 14.46% EER even in cases where the number of samples contributed by the participants is low. The authors worked with data taken from a sample of 40 users who have typed each password 20 times during 4 distinct sessions.

Maxion and Killourhy conducted an experiment of typing a 10-digit number using only the right-hand index finger [26]. A number of 28 users contributed to the experiments by typing 10-digit number on a numeric external keyboard. Capitalising on the random forest classifier and some other statistical techniques to handle the extreme deviations of the collected data, they achieved a detection rate of 99.97% with a corresponding false-alarm rate of 1.51%.

Maiorana et al. introduced a new statistical approach and examined which feature can be used towards differentiating users’ samples best [27]. The authors proposed a keystroke-based verification method with application to mobile devices. They analysed the verification performances achieved when varying several parameters like the time distance between keypress and key release events, the number of enrollment acquisitions, as well as the number of characters contained in the passwords used.

More recently, Saevanee et al. investigated the potential of fusing three different biometric methods, namely behavior profiling, keystroke dynamics, and linguistic profiling, into a multi-modal behavior biometric authentication system [28]. The results they succeeded indicate that such fusion techniques can improve the classification performance with an overall EER of 8%.

2.2. Motion-oriented Keystroke Proposals

The very first work on keystroke using motion sensors appeared in 2011 by Cai and Chen [29]. They proposed a new keylogging scheme based on mobile device motion. They argue that typing (touching) on different locations on the screen causes different vibrations (motion data) which in turn can be used to infer the keys being pressed. Their evaluation shows that the proposed system is able to correctly infer more than 70% of the keystrokes on a numeric virtual keypad when used in landscape mode.

Inspired by the work in [29], several other authors discuss and evaluate their proposals designed with the aim to extract sequences of text entered on virtual keyboards. This is done by taking advantage of only the in-built motion sensors of a smartphone, i.e. the accelerometer and gyroscope. More specifically, Aviv et al. capitalise on the accelerometer sensor as a side-channel to acquire user tap- and gesture-based input, required to unlock mobile devices either when using PIN/password or the Android's graphical password pattern [30]. In controlled settings, while a user is stationary (sitting), their prediction model was able to classify the PIN entered with a percentage up to 43% and the graphical password pattern with 73% within 5 attempts. On the other hand, in uncontrolled settings, while users are pedestrian, their model was able to classify 20% of the PINs and 40% of the graphical password patterns within the same number of attempts. The work by De Luca et al. presented a method to authenticate smartphone users based on the way they interact with the touchscreen. To do so, they cross-evaluated several types of unlock screens and graphical password patterns supported by Android smartphones [31]. Using features such as pressure, size and speed they succeeded an overall accuracy of 77% with a 19% False Rejection Rate (FRR) and 21% False Acceptance Rate (FAR).

Based on their previous work [32], Cai et al. evaluated the way in which device motions caused by keystroking can be exploited in a real attack. More precisely, they developed a prototype attack and applied it on keystroking data and its associated vibrations gathered by a sample of 21 participants. According to their research, the attack remains fruitful even though the accuracy is affected by several factors such as user habits, device dimension, screen orientation, keystroke data and its associated motion information (vibrations), keyboard layout etc. Moreover, Kolly et al. tried to identify users, based on their behavior while playing games on their smartphone [33]. When the user is playing a given game, the touch events occurred on the UI elements are send to a server for offline evaluation analysis. Using the naive Bayes classifier and touch features such as mean hold-time and pressure, the authors were able to successfully identify a given user in a set of 5 individuals with a precision of about 80%.

Miluzzo et al. introduce TapPrints, a framework to infer the exact location one taps and what one types on the touchscreen based on accelerometer and gyroscope sensor readings [34]. In their experiments, engaging 10 participants and three different mobile platforms, the authors demonstrate that TapPrints is able to attain up to 90% and 80% accuracy in inferring tap locations across the display and letters respectively. Xu et al. presented TapLogger a stealth trojan destined to Android platform which is able to log not only the screen lock password but also the numbers entered during a phone call [35]. TapLogger implements two schemes: (a) a tap event detection mechanism to discover and utilise the user's tapping pattern with statistical measurements based on acceleration, and (b) an approach of deducing a tap position with observed gesture changes. In another study, Owusu et al. showed that the accelerometer can be used to extract 6-character passwords in as few as 4.5 trials (median) [36].

In a technical report, Zheng et al. propose a verification system able to identify if the user who is typing a passcode on a touchscreen numeric keypad is the true owner of the mobile device or an impostor [37]. To quantify a given's user tapping behavior four different features have been collected via the corresponding sensors: acceleration, pressure, size, and time. In all the experiments the authors utilised empirical data stemming from both 4-digit and 8-digit PINs. The results showed that their verification system achieves an average EER of 3.65%. Lastly, Rao et al. explored keystroke dynamics as the major factor in behavioral authentication. The authors proposed a method that uses the accelerometer sensor along with the motion one to authenticate the legitimate user of a mobile device [38].

2.3. Proposals based on Gestures

Aviv et al. examined the feasibility of "smudge attacks" on touchscreens for mobile devices [39]. They argued that oily residues (smudges) on the touchscreen surface are one side effect of touches from which frequently used patterns such as a graphical password might be inferred. They focused on Android password patterns and investigated the conditions under which a smudge can be easily extracted. The authors also described the way an ill-motivated person could use the information obtained from a smudge attack to augment the chances of guessing users' touching patterns. Although this work didn't use any on-device software for extracting gestures, it proposed an interesting method for attacking mobile devices with touchscreens.

Angulo and Wastlund studied the use of graphical lock patterns on smartphones and invested on lock pattern biometrics (as a second-factor authentication method) towards identifying users [40]. Using R statistical software, they cross-evaluated 5 machine learning classifiers to identify legitimate users while they form their password graphically. Using the Random Forest classifier, the authors achieved an average EER of approximately 10.39% in the case an impostor already knows the user's secret pattern. The work by Feng et al. proposed a touchscreen-based authentication method for mobile devices coined as Finger-gestures Authentication System using Touchscreen (FAST) [41]. FAST uses the touchscreen and a custom digital sensor collector to gather information about user gestures. According to the authors, FAST is able to support continuous user post-login authentication in a transparent manner. Specifically, their proposed system achieved a FAR, FRR of 4.66% and 0.13% respectively. The work by Sae-Bae et al. introduced a five-finger gesture-based authentication technique able to recognise unique gesture characteristics of an individual [42]. They achieved an accuracy of 90% with only single gestures, while they perceived a significant improvement when multiple gestures were performed in sequence.

Damopoulos et al. introduced the first (software) touchlogger for modern mobile devices [43]. Through experiments the authors showed that touch events collected by a touchlogger can be a reliable and very accurate means of profling the legitimate user(s) of a device. Their results were very promising showing an accuracy in identifying misuses, and thus post-authenticating the user, in an amount that exceeds 99%. Also, using real-use cases they demonstrated the virulent personality of their software when it is used maliciously. Li et al. proposed another user authentication scheme destined to smartphones aiming to continuously re-authenticate the user of the device via the employment of touch gestures [44]. As expected, their system needs to first build the gesture pattern of the owner of the device. After that, it is able to authenticate the current user's finger movements and make an assessment about their authenticity. The system the authors propose is divided into two modules; the training and the re-authentication ones. The former is executed on a PC, while the latter is deployed on a smartphone as a service.

The work by Kambourakis and Damopoulos introduced a fair post-authentication and non-repudiation scheme for mLearning [45]. The proposed scheme can be straightforwardly applied to devices equipped with a touchscreen. Their scheme, based on the biometric modality known as dynamic signature, is able to correctly classify a signature produced in the touchscreen in an amount that exceeds 95% via the use of machine learning algorithms. Yi et al. proposed another user authentication scheme called PassWindow that allows the user to enter their PIN in a secure way. This is done with the help of a window moving on the virtual keypad while utilising an input method based on multimodal sensors [46].

2.4. Discussion

Without doubt, existing authentication methods utilised by modern mobile platforms rely on the "something the user knows" authentication factor, that is, the typical PIN or password. Despite the fact that all the aforementioned researches have significantly contributed to keystroke analysis for mobile devices, several important problems remain unsolved before a full-fledged multi-factor authentication system can be applicable. It is made clear, for example, that the potential of traditional keystroke analysis, as it is utilised on mobile devices with a hardware keyboard, has not yet been assessed for smartphones equipped with a touchscreen. When we refer to traditional keystroke analysis, we mean to utilise only behavioral features, such as Hold-time or Inter-time, that correspond to the way someone interacts with the virtual keyboard. This is exactly where the contribution of the current work lies. Instead, as it is depicted in Fig. 1, most work presented in the literature exploit behavioral features that stem from gestures and readings coming from the motion sensors of the device.

3. TOUCHSTROKING

From the above discussion it is clear that the research community seeks advanced controls for strengthening user authentication in the smartphone realm. Ideally, such controls should be multifactor based as the more the factors, the better security can be attained. As already mentioned, in this paper we consider a two-factor touchstroke user authentication method to discriminate between the legitimate user and intruders. That is, we consider a threat model where everybody may know the valid PIN (first-factor) but the system is able to identify by means of the way and rhythm in which this person interacts with the touchscreen when typing characters, if she is the legitimate user of the device (second-factor). This also means that a training period on how a person - the legitimate user in our case - enters alphanumeric sequences via the touchscreen is needed. As it is explained later in this section, by the term PIN we refer to a password PIN and not to standard 4-digit ones. In this section we provide information on the data collection process, the type and structure of data we are going to analyse as well as on the selected classification methods.

3.1. Authentication scenarios

To assess the efficiency of keystroke dynamics on smartphones, we consider two authentication scenarios widely used in the literature of legacy keystroke anaylsis. This will allow the direct comparison of our results with those obtained by works on keystroke dynamics performed on mobile devices equipped with hardware keyboards. Specifically, in the first scenario, the users need to type down the following password consisted of 10 alphanumeric characters: *7q56n5ll44*. This test was designed to explore if user authentication can be imposed based on typical alphanumeric passwords. Bear in mind that modern smartphones allow the user to replace the 4-digits standard PIN by a password of arbitrary length. Also, the sequence of "*7q56n5ll44*" was selected due to the fact that the keys a user should press are well spread on the virtual keyboard and user's fingers have to move from the one side to the other. In addition, "56", "ll" and "44" are used to capture users' behavior on very close and repeated keys. It has to be noted that no mistakes were allowed and the input process repeated for 12 times for each user. This means that in case a mistake had been made the participant had to repeat that password entering round.

In the second scenario, the users had to type down the phrase "*the quick brown fox jumped over the lazy ghost.*" 12 times each. This particular dummy phrase is considered a standard in keystroke analysis and has been employed by many research works in the field. This is due to its length and the variety of characters it contains. It is therefore generally argued that leads to more accurate typing behavior assessment. It has to be noted that the last word of this particular phrase was initially "dog", but chanced to "ghost" in order to create a phrase including all 26 possible letters of the alphabet. Overall, the phrase consists of 47 characters, including spaces, and a final dot which is used as an indicator to terminate the input process. In this case, the users were allowed to make mistakes under the belief that repeated mistakes could reveal a user's tendency or other specific typing behavior. Figure 2 depicts the UI of the corresponding software as it is displayed on the smartphone during the data collection phase.

3.2. Touchstroke features

In the experiments we employed the following four touchstroking biometric features as input data to machine learning classifiers.

- *Hold-time*: is a commonly used feature in keystroke analysis and refers to the amount of time each key is pressed. It is calculated as the time starting from pressing a given key down until its release.
- *Inter-time*: this is another standard feature widely employed in keystroke dynamics. It refers to the time between the release of a key until pressing the next.

The following two features are specific to this study as, to the best of our knowledge, it is the first time they are considered in the literature of keystroking.

- *Distance*: it represents the distance in pixels between two successively pressed virtual buttons. Note that this feature relies on the size of the soft-keyboard which in turn is affected by the size of the touchscreen.
- *Speed*: it is calculated as the quotient of the *distance* between two successively pressed virtual buttons divided by the *inter-time* for this event to complete. This feature is mostly affected by the size of the touchscreen, but also by the way one holds the device (i.e., with one or both hands) and how many fingers one uses when typing.

3.3. Methodology and data structure

Twenty participants have contributed their data for both scenarios described in section 3.1. All participants were selected to be nearly in the same age of 19 to 21 years old and be owners of an Android smartphone equipped with a touchscreen. Each user executed repeatedly for 12 times both scenarios. This enables the creation of the corresponding behavioral typing (touching) profiles based on the four features described in the previous subsection. Bear in mind that these profiles will be used to train the classifiers.

Two different data analysis methodologies have been evaluated. According to the first one, every pair of successively pressed (virtual) keys are analysed by the classifier. Once all keys of the password/phrase have been entered, the final decision is taken. For the second one, the average values of all the pressed keys per biometric feature are calculated as soon as the user finishes the key typing process. After that, the data are fed to the classifier for the training phase.

Twenty data files have been created per scenario per methodology, i.e., 80 files in total. Each file contains the data of the corresponding legitimate user and those of 17 randomly selected users that represent potential intruders. This means that for each user in the dataset the corresponding data file contains: a) the user's personal data, referred to as normal typing (touching) behavior data, b) 17 other users' data that represent potential intrusive typing behaviors. As noticed, the remaining 2 users' files, out of 19, have been left out of the training body in an effort to examine the efficiency of our system against unknown typing behaviors as well. More specifically, our system needs to be tested against 3 different direct or indirect attack scenarios. First off, the case of a legitimate user rejected by the system, i.e., a false positive, needs to be considered. The second attack has to do with an intruder accepted by the system despite the system has been trained to identify her. The latter addresses the case where a previously unknown person is falsely accepted by the system. Both the last two cases comprise a false negative.

Overall, the data files created for the first methodology contain a number of records depending on the scenario, where each of them corresponds to a vector of related features per key pressing event. Every record contained in the data file is composed of collected features represented by the following quintuplet: {*Speed, Distance, Inter-time, Hold Time, Intruder/Legit*}. On the other hand, the data files that correspond to the second methodology are composed of the average (avg) value of the collected features represented by the following single quintuplet: {*Avg. Speed, Avg. Distance, Avg. Inter-time, Avg. Hold Time, Intruder/Legit*}. The last feature (Intruder/Legit) in the aforementioned quintuplets is the binary representation of the two nominal classes, i.e., if this piece of data belongs to the legitimate user *(False)* or the intruder *(True)*.

An example of such a record is given by the following quintuplet {618.02, 272.23, 0.47, 0.05, TRUE}.

3.4. Software Architecture

A prototype of the proposed authentication system based on touchstroke dynamics has been implemented in Google's Android OS. Figure 3 depicts its overall architecture. The bottom part of Fig. 3 depicts the UI of our keystroke analysis mechanism, essentially the mobile device's screen. The upper part of the same figure provides an abstract representation of the basic sub-mechanisms running in the background. These mechanisms are responsible to create the user profile, store it, and finally authenticate the user.

For constructing keystroke profiles, it is important to collect and analyse the user's typing behavior happening on the mobile device's touchscreen. In most cases, an application cannot acquire touchstrokes unless it is active and receives the focus on the screen. By default, Android does not provide any framework to monitor the keyboard behavior, except the IDs of the pressed keys. This is due to security reasons (for instance, the implementation of a spyware or keylogger would be otherwise easy). However, in our case, features like pressing or releasing of a button, the location in which the user touches the (virtual) key, and the timestamp a touch action took place are required for the creation of such a behavior profile. To tackle the aforementioned limitation it was necessary to design and implement our own custom keyboard views, depicted in the bottom side of Fig. 3, and connect them with the appropriate key listeners within our application. Furthermore, to be able to listen to touching events stemming from the virtual keyboard, the implementation of *OnKeyboardActionListener*, *OnKeyListener* and *OnTouchListener* Java interfaces as defined in the Android API and their abstract methods was important. This way we were able to take advantage of the features, as they are described in section 3.2, deriving from user's keystrokes, and use them to build this user typing (touch) profile (see Fig. 3, cases 1-3).

Although our method can be applied only to applications that utilise the custom keyboard view, it is straightforward to provide a complete solution using the Cydia Substrate library for overwriting any native Android keyboard view [47]. Specifically, we argue that the proposed mechanism can be straightforwardly adapted for use with a variety of applications (e.g., e-banking apps, mobile web browsers) that support any type of soft-keyboard and running on virtually any device equipped with a touchscreen. Mobile devices with different screen size or keyboard can also be supported, as soon as a new user typing profile is created for them. The exploration of ways a user's typing profile can be automatically migrated to any soft-keyboard or touchscreen size is also an interesting research direction for future work.

Moreover, machine learning classifiers need to run directly on the device for being able to create the user's typing behavior profile, learn from it, and finally attempt to authenticate the user (see Fig. 3, case 4). The well-known machine learning software package, namely Waikato Environment for Knowledge Analysis (Weka), has been employed as the classification engine for our solution (see Fig. 3, case 5). As illustrated in Fig. 4, our mechanism, consists of two main analysis sub-systems; the *Enrolment Process*, which allows a user to create their typing profile and train the classifiers with it, and the *Authentication Process*, which is in charge of examining the legitimacy of a given user. As soon as the user's behavior profile is created, and the classifier is trained, the user profile is stored in a local database.

Upon user authentication, say, the user enters their password to unlock the device, the authentication system running in the background will feed the *Classifier* with the legitimate user's profile as retrieved from the database, plus the current user typing (touching) behavior in terms of a vector of features as discussed in sections 3.2 and 3.3. Then, the Classifier will make an assement and categorise the user into one of the two possible classes; intruder or non-intruder. It will also take the appropriate action depending on the case.

4. EVALUATION

In an effort to select the most appropriate machine learning algorithm per methodology to be used as the classification engine for the proposed system we conducted some preliminary classification experiments considering three popular classifiers; Random Forest, KNN, and MLP. As reported in [48], these three classifiers seem to be the most prominent in terms of performance when mobile devices are concerned. Putting it another way, one needs to select not only the most effective algorithm in terms of classification, but also the one that is able to execute fast on a smartphone having limited CPU and memory resources. Thus, to perform these initial experiments we provided the aforementioned classifiers with an upper bound of 512 MB memory, which is a common value across modern smartphones. The results showed that MLP was not able to run with this memory restriction, so it was rejected outright. On the other hand, as discussed further in this section, Random Forest and KNN obtained favorable results, and thus constitute promising options for our classification engine running directly on the device.

4.1. Effectiveness

Legacy biometric systems effectiveness analysis makes use of two error rates, namely False Acceptance Rate (FAR) in which an intruder is accepted by the system, and False Rejection Rate (FRR) in which the authorised user is rejected by the system. In addition, a third metric known as Equal Error Rate (EER) is generally employed to examine the performance of similar to ours biometric systems. Specifically, EER is a kind of percentage rate which both accepts and rejects errors as equals ($EER=(FAR+FRR)/2$).

This metric is employed to quantify the detection accuracy by a single number.

Tables I and II, summarise the results logged for the previously mentioned metrics per scenario, per methodology and per classifier. It can easily observed that the best results for the first scenario have been obtained when using Random Forest and the first methodology. For the second scenario KNN scores better when the second methodology is considered. More specifically, the average FAR%, FRR% and EER% values for the first and second scenario are *(12.5, 39.4, 26)* and *(23.7, 3.5, 13.6)* respectively.

Having the above results, one can argue that for the second scenario, i.e., when the user authenticates itself by entering a passphrase, the second methodology along with the KNN classifier seem to be the best choice towards implementing an authentication system based on touchstroke dynamics. In this case, the system accepts an intruder with a percentage equal to 23.7%, while rejects the legitimate user with that of 3.5%. Note that the latter percentage is the most important here as it is directly associated with the legitimate user. So, while an impostor would gain access after about 4 attempts, the legitimate user would be rejected rarely. The results obtained for the first scenario follow the opposite norm as it seems that the legitimate user is rejected with 39.4%, while an intruder is accepted with a percentage of 12.5%. These results are not come as a surprise because simply they verify the rule that the more the characters in the passphrase the better the classification outcome. As a result, some very low percentages regarding the error rates spotted by some studies [23, 24, 26] do come unexpectedly. In any case, in terms of EER, the values we obtained are directly comparable with those reported by other studies in the area of keystroking. This situation is shown in the leftmost part of Fig. 1 which summarises in a chronological order all keystroking works already presented in section 2 along with their EER. Note that comparison with works related to motion- or gesture-based proposals is in fact aimless due to the different methodologies and biometric features used by these studies to observe keystroking events. In short, as it can be easily observed from Fig. 1, most EER values fluctuate between 8 and 14.4%. Lastly, regarding the methodology, it can be said that the longer the passphrase, the better the second methodology scores. On the other hand, the first methodology seems to perform better when a shorter passphrase is in use.

4.2. Performance

Although high efficiency remains the first goal when designing a security system, performance metrics are also of major importance. By the term performance we refer to the time period a security mechanism needs to reach a decision, but also the computational and memory resources it consumes. While modern smartphones do not afford considerable CPU and memory resources, at least when compared to those of former generations, performance still remains decisive. Past researches on keystroke dynamics and closely related literature as they discussed in section 2 systematically neglect to occupy themselves with performance. This is because while the data were collected on the mobile device, the experiments have been conducted on desktop systems. So, an important contribution of this work is that evaluates the proposed mechanism directly on the smartphone in terms of speed, CPU and memory. The experiments have been conducted for both classifiers (RF, KNN) considering also both scenarios and methodologies. The device used throughout the experiments was a Sony Ericsson Xperia ray equipped with 1 GHz CPU processor, 512 MB RAM, 3.3 inches touchscreen and the Ice Cream Sandwich Android OS. During the experiments only our application was running on the smartphone. However, keep in mind that due to the Android multitasking environment all metrics regarding the classification time, CPU and memory consumption may have been negatively affected by the various OS native services running in the background of the device.

Figures 4 and 5 contain all performance metrics as they recorded on the smartphone device. It is obvious that machine learning algorithms have consumed 100% of CPU during the classification procedures. For the training ones this percentage varies between 91 and 100%. On the other hand, memory consumption fluctuates between 67.5 and 80.2% depending on the methodology, scenario and classification algorithm used. From Table III it can be inferred that the first methodology requires substantially more memory to execute. This is however expected as this methodology produces far more data than the second one. It is worth noting here that this particular smartphone shows 52% of its memory to be occupied just after rebooting. Table III offers an aggregated comparative view of the various classification times in seconds. The average percentage metrics for CPU and memory consumption are included in the same table as well. From the table it can be brought forward that both algorithms were able to execute directly on the device, authenticating the user in less than a second in all cases. As expected, the training phase was the one with the longest duration, while the classification phase utilised 100% of CPU. Random Forest's training phase was longer compared to that of KNN's, while for the testing phase the opposite behavior was observed. The first methodology seems to consume more memory irrespective of the classifier employed. Generally, we can argue that Random Forest consumes more time during the training phase but is faster in producing the decision on an authentication attempt. Regarding the second scenario and methodology, KNN seems to be the optimal choice as it scores best in all efficiency and performance tests.

5. CONCLUSIONS & FUTURE WORK

Mobile devices have evolved and experienced an immense popularity over the last few years. Nevertheless, this growth has exposed smartphones to an augmenting number of security threats. It is thus for sure that despite the variety of peripheral protection mechanisms described in the literature and the authentication and access control techniques imposed by the OS of such devices, integral protection against advanced intrusions cannot be adequately enforced. The review of the smartphone security ecosystem on biometrically-inspired methods included in this article reveals that while a critical mass of works have been done during the last few years, there is still need for advanced protection mechanisms with particular focus on user (post)authentication procedure.

Compelled by this fact, in this paper, we make a first attempt to assess keystroke dynamics in the realm of smartphones equipped with a touchscreen. So, by paraphrasing the previous standard term we can say that this work is involved with "touchstroke dynamics". By the use of legacy scenarios used in keystroke analysis but also via the exploration of novel biometric features and methodologies we concluded that touchstroking has significant potential in designing enhanced authentication systems destined to future smartphones. Specifically, when considering the best results achieved during the experiments, one can argue that the FAR value of 3.5 is very promising. The same applies for the minimum EER value of 12.5. While these results are not a breakthrough, they are very close - and some times better - to those recorded by previous studies in the area of keystroke dynamics for mobile devices having hardware keypads.

Another contribution of this work is that it reports on test results contacted directly on an Android smartphone. Thus, it is expected that the outcomes of the current paper may serve as a pilot for future studies in the field. It is however certain that further research in this area is needed to better assess the power of this newly introduced biometric modality. So, as a future work, we would like to extend this study by (a) considering more classifiers, (b) taking into account advanced scenarios, methodologies and features, (c) designing automatically adjusted user typing profiles that hopefully can be applicable to any soft-keyboard or touchscreen size, (d) exploring the possibility of of touchstroking as a mechanism to support continuous authentication, and (e) increasing the number of and diversity participants in future experiments to better assess the outcomes associated with this particular behavioral trait.

REFERENCES

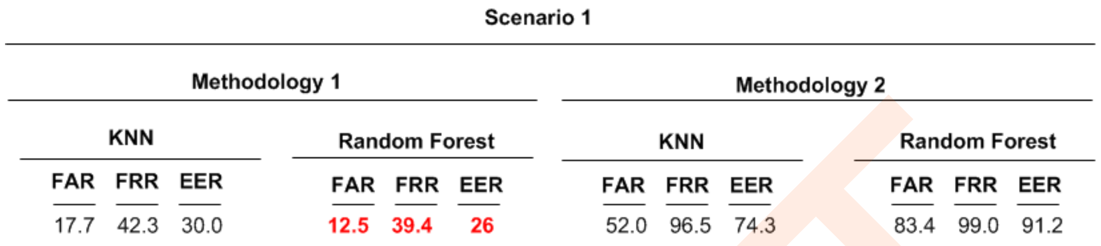
1. Lookout. State of mobile security 2012 2012. https://www.lookout.com/resources/reports/state-of-mobile-security-2012.
2. Lookout Mobile Security. DroidDream. http://blog.mylookout.com/droiddream/ 2012.
3. Damopoulos D, Kambourakis G, Gritzalis S. iSAM: An iphone stealth airborne malware. *Future Challenges in Security and Privacy for Academia and Industry*, *IFIP Advances in Information and Communication Technology*, vol. 354, Camenisch J, Fischer-Hubner S, Murayama Y, Portmann A, Rieder C (eds.). Springer Berlin Heidelberg, 2011; 17–28.
4. Damopoulos D, Kambourakis G, Anagnostopoulos M, Gritzalis S, Park J. User privacy and modern mobile services: are they on the same path? *Personal and Ubiquitous Computing* 2012; :1–12.
5. Chavez A. A jailbroken iphone can be a very powerfull weapon in the hands of an attacker. *Technical Report*, Purdue University, Calumets CIT Department 2008.
6. An Garda Siochana. Launch of leaflet - theft of smart phones in dublin 2012. http://garda.ie/Controller.aspx?Page=10995.
7. Plateau. Smartphone safety tips 2011. http://www.plateautel.com/wireless_stolen_phones.asp.
8. Mobile Insurance. The underground world of mobile phone theft 2013. http://www.mobileinsurance.co.uk/images/mobile_phones_theft_tube.jpg.
9. Han J, Kywe S, Yan Q, Bao F, Deng R, Gao D, Li Y, Zhou J. Launching generic attacks on ios with approved third-party applications. *Applied Cryptography and Network Security*, *Lecture Notes in Computer Science*, vol. 7954, Jacobson M, Locasto M, Mohassel P, Safavi-Naini R (eds.). Springer Berlin Heidelberg, 2013; 272–289.
10. Berkman O, Ostrovsky O. The unbearable lightness of pin cracking. *Financial Cryptography and Data Security*, *Lecture Notes in Computer Science*, vol. 4886, Dietrich S, Dhamija R (eds.). Springer Berlin Heidelberg, 2007; 224–238.
11. Schulz T. Using the keystroke-level model to evaluate mobile phones. *Proceedings of the 31st Information Systems Research Seminaria - IRIS 31*, Scandinavia, 2008.
12. Park YS, Han SH, Park J, Cho Y. Touch key design for target selection on a mobile phone. *Proceedings of the 10th international conference on Human computer interaction with mobile devices and services - MobileHCI*, ACM: New York, 2008; 423–426.
13. Feher C, Elovici Y, Moskovitch R, Rokach L, Schclar A. User identity verification via mouse dynamics. *Information Sciences* 2012; **201**(0):19 – 36. Elsevier.
14. Stefan D, Shu X, Yao D. Robustness of keystroke-dynamics based biometrics against synthetic forgeries. *Computers & Security* 2012; **31**(1):109 – 121. Elsevier.
15. Vuagnoux M, Pasini S. Compromising electromagnetic emanations of wired and wireless keyboards.

Proceedings of the 18th conference on USENIX security symposium - SSYM'09, USENIX Association, 2009; 1–16.

16. Adhikary N, Shrivastava R, Kumarl A, Verma SK, Bag M, Singh V. Battering keyloggers and screen recording software by fabricating passwords. *International Journal of Computer Network and Information Security - (IJCNIS)* 2012; **4**(5):13–21.
17. Clarke NL, Furnell SM, Lines B, Reynolds P. Subscriber authentication for mobile phones using keystroke dynamics. *Proceedings of the 3rd International Network Conference - INC*, UK, 2002; 347–355.
18. Clarke NL, Furnell SM. Authenticating mobile phone users using keystroke analysis. *International Journal of Information Security* 2007; **6**(1):1–14. Springer.
19. Clarke N, Furnell S. Advanced user authentication for mobile devices. *Computers & Security* 2007; **26**(2):109–119.
20. Karatzouni S, Clarke NL. Keystroke analysis for thumb-based keyboards on mobile devices. *New Approaches for Security, Privacy and Trust in Complex Environments - IFIP International Federation for Information Processing*. Springer Boston, 2007; 253–263.
21. Buchoux A, Clarke NL. Deployment of keystroke analysis on a smartphone. *Proceedings of the 6th Australian Information Security Management Conference - SECAU*, Western Australia, 2008; 40–47.
22. Saevanee H, Bhattarakosol P. Authenticating user using keystroke dynamics and finger pressure. *Proceedings of the 6th IEEE Consumer Communications and Networking Conference*, IEEE: Ireland, 2009; 1–2.
23. Zahid S, Shahzad M, Khayam S, Farooq M. Keystroke-based user identification on smart phones. *Proceedings of the 12th International Symposium on Recent Advances in Intrusion Detection - RAID '09*, Springer-Verlag, 2009; 224–243.
24. Hwang S, Cho S, Park S. Keystroke dynamics-based authentication for mobile devices. *Computers & Security* 2009; **28**(1-2):85 – 93. Elsevier.
25. Campisi P, Maiorana E, Bosco ML, Neri A. User authentication using keystroke dynamics for cellular phones. *IET Signal Processing - Special Issue on Biometric Recognition* 2009; **3**(4):333–341.
26. Maxion R, Killourhy K. Keystroke biometrics with number-pad input. *Dependable Systems and Networks (DSN), 2010 IEEE/IFIP International Conference on*, 2010; 201–210.
27. Maiorana E, Campisi P, Gonzlez-Carballo N, Neri A. Keystroke dynamics authentication for mobile phones. *Proceedings of the 2011 ACM Symposium on Applied Computing - SAC*, ACM: USA, 2011; 21–26.
28. Saevanee H, Clarke NL, Furnell SM. Multi-modal behavioural biometric authentication for mobile devices. *Proceedings of the Information Security and Privacy Research, IFIP Advances in Information and Communication Technology - IFIP AICT*, Springer Boston, 2012; 465–474.
29. Cai L, Chen H. Touchlogger: Inferring keystrokes on touch screen from smartphone motion. *Proceedings of the 6th USENIX Workshop on Hot Topics in Security - HotSec*, 2011.
30. Aviv AJ, Sapp B, Blaze M, Smith JM. Practicality of accelerometer side channels on smartphones. *Proceedings of the 28th Annual Computer Security Applications Conference*, ACSAC '12, ACM: New York, NY, USA, 2012; 41–50.
31. De Luca A, Hang A, Brudy F, Lindner C, Hussmann H. Touch me once and i know it's you!: implicit authentication based on touch screen patterns. *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems*, CHI '12, ACM, 2012; 987–996.
32. Cai L, Chen H. On the practicality of motion based keystroke inference attack. *Proceedings of the 5th international conference on Trust and Trustworthy Computing*, TRUST'12, Springer-Verlag, 2012; 273–290.
33. Kolly SM, Wattenhofer R, Welten S. A personal touch: recognizing users based on touch screen behavior. *Proceedings of the Third International Workshop on Sensing Applications on Mobile Phones*, PhoneSense '12, ACM, 2012; 1–5.
34. Miluzzo E, Varshavsky A, Balakrishnan S, Choudhury RR. Tapprints: your finger taps have fingerprints. *Proceedings of the 10th international conference on Mobile systems, applications, and services -MobiSys '12*, ACM, 2012; 323–336.
35. Xu Z, Bai K, Zhu S. Taplogger: inferring user inputs on smartphone touchscreens using on-board motion sensors. *Proceedings of the 5th ACM conference on Security and Privacy in Wireless and Mobile Networks - WISEC '12*, ACM; 113–124.
36. Owusu E, Han J, Das S, Perrig A, Zhang J. Accessory: password inference using accelerometers on smartphones. *Proceedings of the 12th Workshop on Mobile Computing Systems and Applications - HotMobile '12*, ACM, 2012.
37. Zheng N, Bai K, Huang H, Wang H. You are how you touch: User verication on smartphones via tapping behaviors. *Technical Report*, College of William & Mary Department of Computer Science 2012.
38. Rao K, Anne V, Sai Chand U, Alakananda V, Navya Rachana K. Inclination and pressure based authentication for touch devices. *ICT and Critical Infrastructure: Proceedings of the 48th Annual Convention of Computer Society of India- Vol I, Advances in Intelligent Systems and Computing*, vol. 248, Satapathy SC, Avadhani PS, Udgata SK, Lakshminarayana S (eds.). Springer International Publishing, 2014; 781–788.

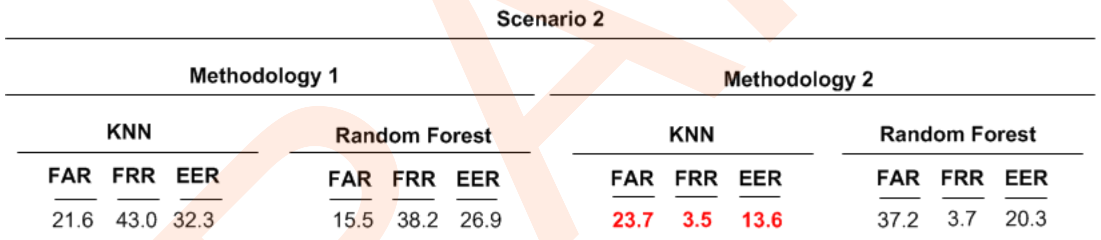
39. Aviv A, Gibson K, Mossop E, Blaze M, Smith J. Smudge attacks on smartphone touch screens. *Proceedings of the 4th USENIX Workshop On Offensive Technologies - WOOT*, 2010.
40. Angulo J, Wastlund E. Exploring touch-screen biometrics for user identification on smart phones. *Privacy and Identity Management for Life*, *IFIP Advances in Information and Communication Technology*, vol. 375, Camenisch J, Crispo B, Fischer-Hubner S, Leenes R, Russello G (eds.). Springer Berlin Heidelberg, 2012; 130–143.
41. Feng T, Liu Z, Kwon KA, Shi W, Carbunar B, Jiang Y, Nguyen N. Continuous mobile authentication using touchscreen gestures. *2012 IEEE Conference on Technologies for Homeland Security (HST)*, 2012; 451–456.
42. Sae-Bae N, Ahmed K, Isbister K, Memon N. Biometric-rich gestures: a novel approach to authentication on multi-touch devices. *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems*, CHI '12, ACM, 2012; 977–986.
43. Damopoulos D, Kambourakis G, Gritzalis S. From keyloggers to touchloggers: Take the rough with the smooth. *Computers & Security* 2013; **32**(0):102 – 114.
44. Li L, Zhao X, Xue G. Unobservable re-authentication for smartphones. *Proceedings of the NDSS Symposium 2013 - NDSS2013*, IEEE.
45. Kambourakis G, Damopoulos D. A competent post-authentication and non-repudiation biometric-based scheme for m-learning. *Proceedings of the 10th IASTED International Conference on Web-based Education (WBE 2013)*, ACTA Press, 2013; 821–827.
46. Yi H, Piao Y, Yi J. Touch logger resistant mobile authentication scheme using multimodal sensors. *Advanced in Computer Science and its Applications*, *Lecture Notes in Electrical Engineering*, vol. 279, Jeong HY, Yen NY, Park JJJH (eds.). Springer Berlin Heidelberg, 2014; 19–26, doi:10.1007/978-3-642-41674-3_4.
47. SaurikIT, LLC. Cydia Substrate. http://www.cydiasubstrate.com/ 2013.
48. Damopoulos D, Menesidou SA, Kambourakis G, Papadaki M, Clarke N, Gritzalis S. Evaluation of anomaly-based ids for mobile devices using machine learning classifiers. *Security and Communication Networks* 2012; **5**(1):3–14.

Table I. Results regarding the first scenario (best scores are in bold red)



Scenario 1											
Methodology 1						Methodology 2					
KNN			Random Forest			KNN			Random Forest		
FAR	FRR	EER	FAR	FRR	EER	FAR	FRR	EER	FAR	FRR	EER
17.7	42.3	30.0	**12.5**	**39.4**	**26**	52.0	96.5	74.3	83.4	99.0	91.2

Table II. Results regarding the second scenario (best scores are in bold red)



Scenario 2											
Methodology 1						Methodology 2					
KNN			Random Forest			KNN			Random Forest		
FAR	FRR	EER	FAR	FRR	EER	FAR	FRR	EER	FAR	FRR	EER
21.6	43.0	32.3	15.5	38.2	26.9	**23.7**	**3.5**	**13.6**	37.2	3.7	20.3

Table III. CPU and Memory performace results (the bold red values correspond to those from tables I, II also in bold red)

Methodology	Scenario	KNN						Random Forest					
		Time (sec)		Memory (%)		CPU (%)		Time (sec)		Memory (%)		CPU (%)	
		Train	Test	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test
1	1	1.17	0.06	79.0	79.3	98.0	100	**2.43**	**0.07**	**79.4**	**79.7**	**96.5**	**100**
	2	3.78	0.10	79.0	79.2	95.8	100	9.00	0.06	80.0	80.2	94.5	100
2	1	0.52	0.02	68.1	68.3	100	100	0.83	0.01	67.7	67.8	100	100
	2	**0.03**	**0.58**	**67.5**	**67.6**	**100**	**100**	0.32	0.01	66.9	66.9	91.0	100

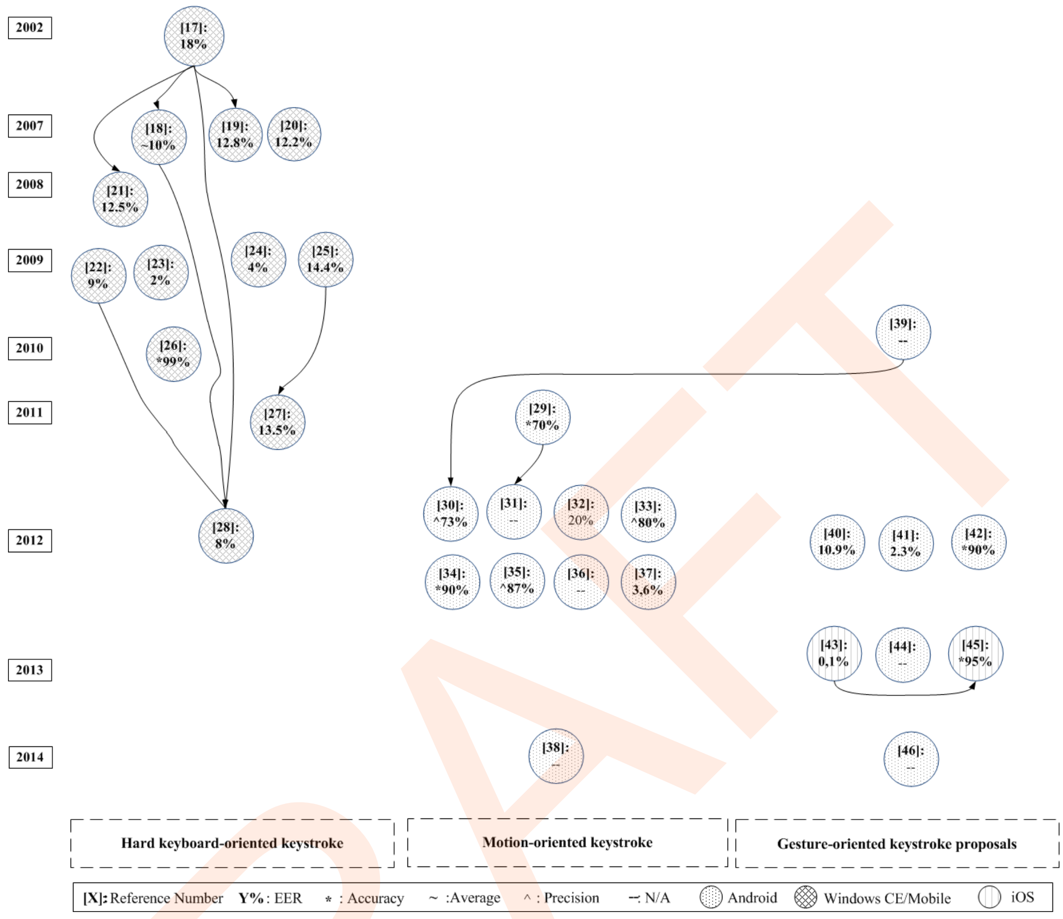


Figure 1. Major approaches related directly or indirectly to keystroke dynamics in chronological order.

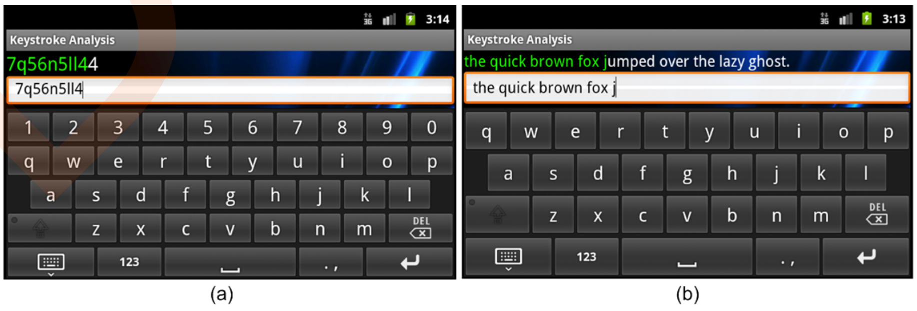


Figure 2. UI corresponding to the first (a) and second (b) scenario

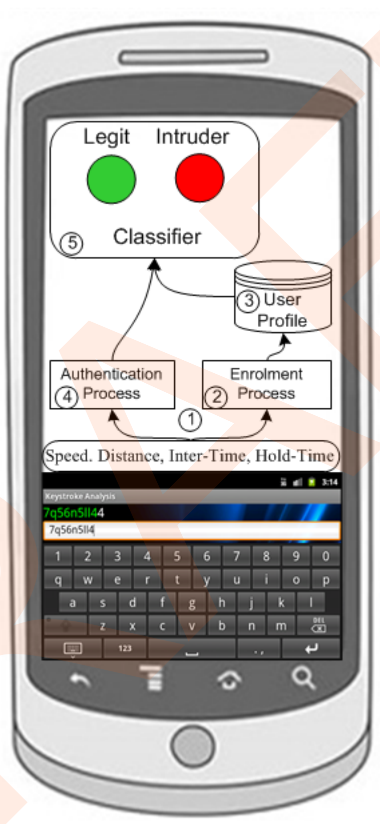


Figure 3. Abstract architecture of the proposed touchstroke solution

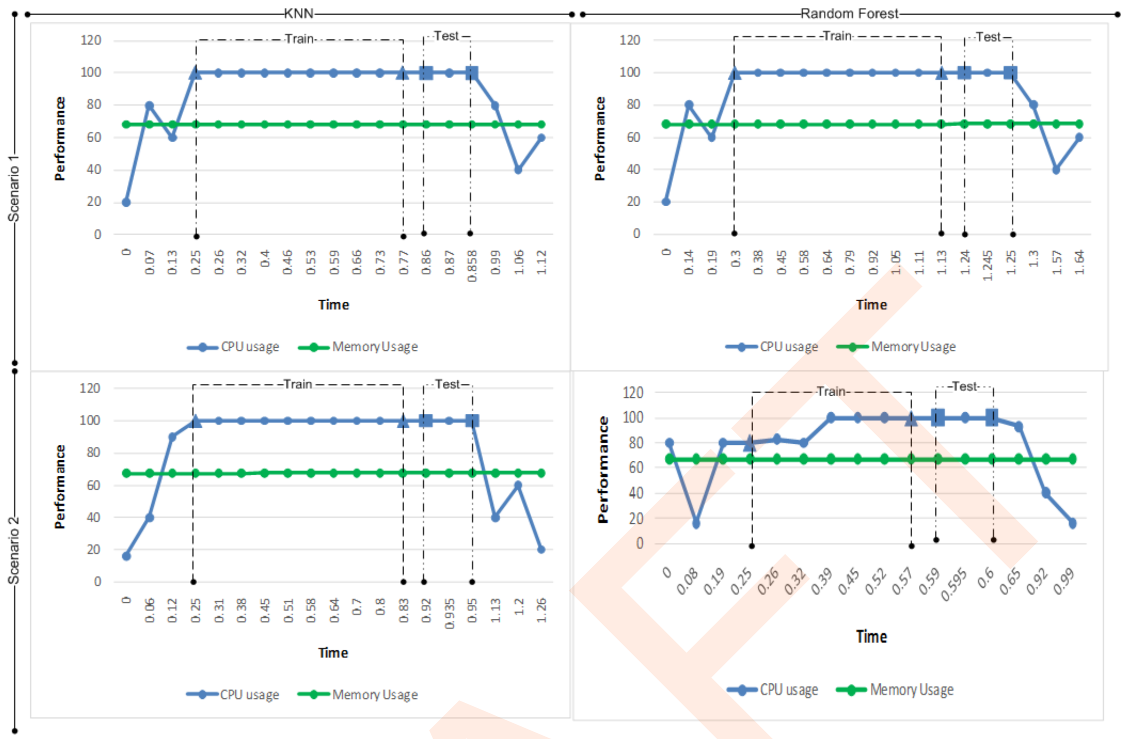


Figure 4. Performance results for the first methodology

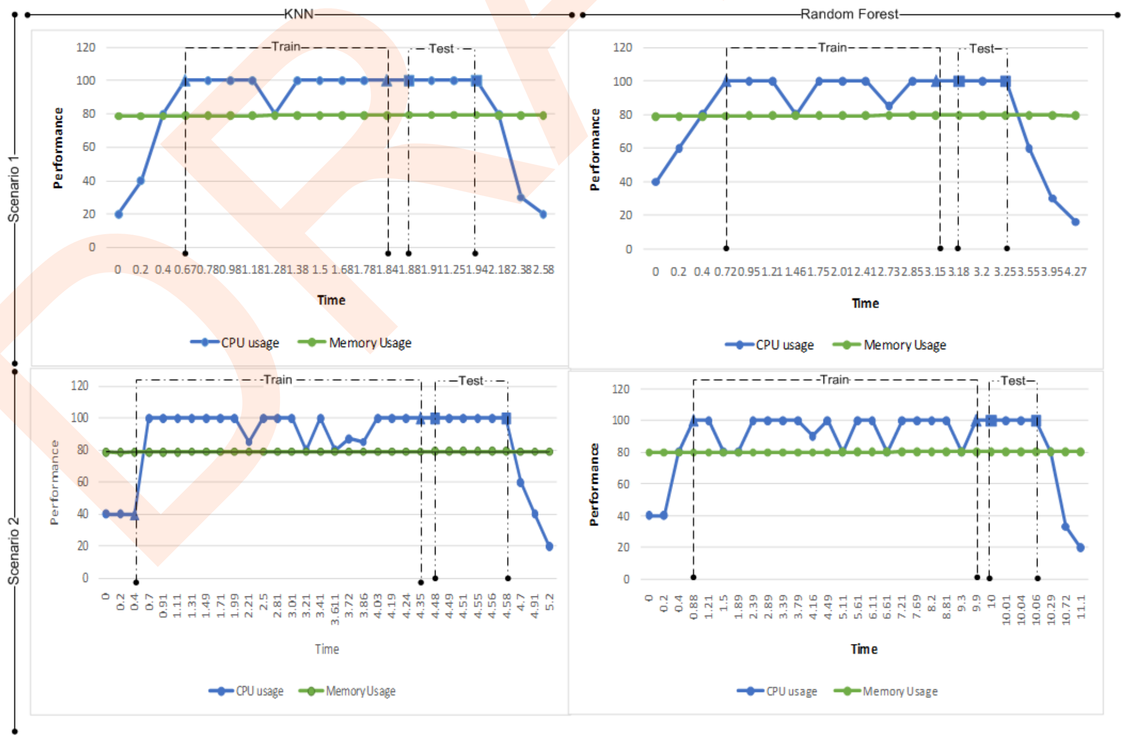


Figure 5. Performance results for the second methodology