University of Twente
Department of Computer Science

Master Thesis

Contextual Authentication

Using Mobile Phone Movements to Authenticate Owners Implicitly

Master Thesis of:
Yazan Badin

Graduation Committee:
Dr. Maarten Everts (University of Twente)
Dr. Andreas Peter (University of Twente)
Dr. Arjan Jeckmans (AET Europe)
Bert Smid, MBT (AET Europe)

2016

Acknowledgments

I would like to offer my sincerest gratitude to my supervisors Dr. Maarten, Dr. Arjan, and Mr. Bert for their superb support and guidance. This thesis would have never seen the light without them. My greatest thanks go to my family for their unconditional love. I would also like to thank my amazing friends for helping me collecting data, testing my implementations, and proofreading. Special thanks for all the people at AET for their kindness.

Abstract

Current security approaches used on mobile phones such as PIN and passwords have been proven to have weaknesses. These weaknesses include susceptibility to shoulder surfing attacks and to smudge attacks. These kinds of security mechanics work on an all-or-nothing basis, meaning that once a password is entered correctly, the user has access to everything. Another weakness comes from the discrepancy between security and usability; users tend to intentionally use weak passwords for the sake of usability. The reason for this comes from the fact that the usage pattern of mobile devices is characterized by small bursts of activity. This in turn leads the users to type their passwords/PIN every time they want to use the phone. Moreover, users tend to not use a password or PIN at all in some occasions. Advanced techniques like face and fingerprint recognition can also be circumvented, they hinder usability, and they might need special hardware. For example, fingerprint recognition needs a special sensor and a fingerprint can be reconstructed relatively easy from a surface or even from a photograph. Then, it can be used to access any fingerprint-secured device.

The solution to these problems is authenticating owners of mobile phones implicitly using context. Implicitly means not requiring the user to perform any additional task, but instead using the available data a mobile phone produces when users interact with their phones. Context can include rich information such as location and device fingerprints, but the interesting type of context is behavior analysis, which takes advantage of the relation between a phone and its owner. Hence, contextual authentication aims to increase both security and usability by authenticating users through the way they interact with their phones rather than requiring them to perform specific tasks, such as entering a password.

The purpose of this work is to explore the possibility of providing continuous and implicit authentication from owner to mobile phone while assuring high accuracy that can be acceptable in practical situations. We focus on utilizing movements that are natural to phone usage. Natural movements happen when users are interacting with their phones. This means that the solution should not require users to perform any additional tasks to authenticate themselves, they should only use their phones. Movements that are natural to the phone usage that we will investigate are (1) the way users pick up their phones from a table, and (2) the micro-movements of the phone when users interact with them. Both mentioned movements serve one of the established goals which is "implicit authentication". The micro-movements serve another goal which is "continuous authentication". Continuous authentication means that the user is always being authenticated in the background, especially when the user wants to access a critical function such as a banking app. An important aspect of the pick-up motion is that it usually precedes any other interaction, meaning that the user needs to first pick up the phone before starting to use it. Thus, the pick-up motion can serve as a first line of defense. On the other hand, the micro-movements authentication can serve as a second, and continuous, line of defense.

An important objective to accomplish is to extract features from the collected data (such as average, standard deviation, amplitude, etc) that can be informative of this data and then select some of these features that give a distinction to the two motions and can be utilized to identify the owner of a mobile device. Another important objective is to choose the right classification algorithm that suits these two types of movements. In addition, classification algorithms usually have specific parameters that need to be chosen carefully to get the best out of the produced model. The last objective is to build a pattern recognition process that can correctly detect a pick-up motion (regardless of the user) at the right time, otherwise the pick-up authentication mechanism would be useless.

The first step of this work was to build a simple Android app to collect sensor data (accelerometer and orientation sensor). A group of participants were asked to pick up the phone from a table and type on the phone multiple times while the app collected the acceleration on X, Y, and Z axes and the orientation angles Yaw, Pitch, and Roll. This data then was manually analyzed and it was apparent that there are

noticeable differences between different users. Then, machine learning was used to build a classification model. The model included two classes: owner and non-owner. When new data is supplied to the model, it can predict to which class the current user belongs. To this end, two algorithms were used: Dynamic Time Warping (DTW) for the pick-up motion, and Support Vector Machine (SVM) for the micro-movements. Raw sensor data were used with DTW to build a model. In the case of SVM, multiple features were extracted from sensor data to help build the model. Subsequently, a test was performed to evaluate the accuracy of the model. For this purpose, a prototype was created (Android app) to perform the tests. During the test, participants would pick up the phone or type, and the app would display the algorithm decision. The results showed that those two motions can successfully be used to differentiate between the owner of the device and intruders. The pick-up motion achieved 3.3% FRR and 0% FAR. The micro-movement achieved 9.5% FRR and 9.2% FAR for the Polynomial kernel and 6.8% FRR and 12.3% FAR for the Sigmoid kernel.

Contents

List of Figures

List of Tables

Introduction 1

The increased usage of information technology has resulted in organizations and individuals having more interactions with physical and electronic assets. These assets usually contain sensitive information that needs to be protected. This sensitive information includes personal data ranging from calender entries to health-related data. In addition, these devices are used to conduct financial operations such as shopping online or even executing bank transactions. As a result, reliable authentication is required more than ever. Despite the huge spending on security solutions, surveys have shown that the number of businesses suffering from security breaches is increasing rapidly [46]. The problem resides in the fact that the current popular authentication tools such as passwords have proved to be susceptible to attacks. Furthermore, the two worst used passwords, according to a study of 2 million exposed passwords, are "123456" and "password" [61]. Another example is shoulder surfing attacks and smudge attacks that can be performed on mobile devices. More advanced techniques such as fingerprint and face recognition introduce their own weaknesses; they need special hardware and can be circumvented [31, 48]. Another possibility is the usage of One Time Passwords (OTPs), however, using OTP implies the usage of additional hardware that incurs additional cost which limits their wide usage [45]. A study conducted by Lawless Research in 2016 [30] shows that 36% of companies foresee that they will do away with passwords in 1 to 4 years, and another 36% predict they will no longer use them in 5 to 9 years. Moreover, the study shows that the vast majority of security professionals no longer trust the password to do its job. In addition to security, usability[1] is an important aspect needed in personal devices. That being said, security and usability usually compete against each other. For example, if users are not required to input passwords to access their email accounts then it would be very easy for users but it would have no degree of security. Therefore, a balance must be found between security and usability. Especially since users do not only demand their devices to be more secure, but also more user-friendly.

1.1 Authentication and Current Approaches

Authentication is the security practice of affirming that an entity is in fact who or what it stated to be. The entity might be a person or a nonperson such as a router [38]. Essentially, authentication of persons can be categorized into three different types depending on what the users use to authenticate themselves:

[1] Usability is a quality attribute that assesses how easy a device or an interface are to use [60].

- Knowledge Based: Something the user knows. This can be for example a personal identification number (PIN), or a password.
- Possession Based: Something the user has. This typically is a hardware device such as a security token, a smart card, or a mobile device.
- Biometric Based: Something the user is. Usually, this is related to human characteristics. There are two types of biometric data: (1) physiological such as fingerprint, iris, face, and retina, and (2) behavioral such as keystroke dynamics, and gait[2] analysis.

Authentication can also be categorized based on the number of techniques used [38]:

- Single-factor Authentication: Authentication of one entity to another using only one authentication technique.
- Multi-factor Authentication: The use of two or more independent and different authentication techniques together. These techniques must belong to different categories. For example, using a password and fingerprint is a correct use of multi-factor authentication, but using fingerprint and face recognition is not.

Finally, authentication can be categorized into two broad groups: Explicit and Implicit authentication. Explicit authentication is when users intentionally perform a task to authenticate themselves such as entering a PIN. On the other hand, implicit authentication does not require the user to input specific data. In this case, the device recognizes its user without demanding a password or a fingerprint scan. One way of authenticating users implicitly is using contextual authentication. This means authenticating users with data that is generated by a device when the device is being used. The next section will define contextual authentication, explain its importance, and describe the kind of data that can be used.

1.2 Contextual Authentication

Contextual authentication aims to provide authentication using context: authenticating entities implicitly instead of explicitly asking them to authenticate. Context can include rich information such as:

- Location: Authentication based on context can, for example, compare the current geographic location of the user with known good or bad locations and take a decision accordingly.
- Device Fingerprinting: In this method, a device is registered first and fingerprinted. This fingerprint can include data such as installed fonts, browser plug-ins, and screen resolution [56]. On subsequent login attempts, the system can check the device which is trying to access a resource against the stored fingerprint.
- Behavior Analysis: In contrast to physiological characteristics, which can be used as an explicit method for authentication, behavioral characteristics can be used to design context-based authentication systems. These systems can build profiles for users' behavior when interacting with a device such as mouse movements and keystroke dynamics. When a user tries to authenticate, the system can check whether the current user's captured data is within acceptable parameter boundaries.

[2] Gait is the pattern of movement of the limbs of animals, including humans, during locomotion over a solid substrate [62].

Key points about contextual authentication are exploiting the idea that humans are creatures of habit and the relation between users and their devices. Habits, such as the time of the day a user leaves for work and the route a student takes to school, can be used to help determine if the device is being carried by its owner. Contextual authentication aims to:

- Minimize user effort by using context data to authenticate the user. This would decrease the amount of times a user needs to explicitly authenticate.
- Offer more fine-grained protection, especially when users attempt to access sensitive information such as calendar entries or critical tasks such as a money transfer.
- Be less dependent on secret knowledge. As mentioned earlier, users tend to use weak passwords for their devices or not use them at all. Context does not require the user to memorize any additional information.
- Resist observation. Attackers may be able to steal users PIN or pattern (shoulder surfing attacks or smudge attack) but it is considerably hard to steal or imitate the way users interact with their phones.

1.2.1 Why Contextual Authentication?

The most important aspect of using contextual authentication is that it can strike a balance between security and usability. As mentioned previously, classical security mechanisms have weaknesses. Passwords, which are used to access almost all of our online accounts, are used by humans, and humans are the weakest link in IT security. Passwords can be stolen by phishing attacks or by guessing attacks. Moreover, a project done by Larry Ponemon showed that when users are forced to use 15-character passwords, they usually write it down on a post-it note [54]. As for advanced techniques, such as fingerprints, they as well have flaws. For example, a photo of a fingerprint can be used along with cheap materials to reconstruct the fingerprint [49]. In the case of face recognition, this method of authentication can also be deceived using a photo even with systems that require the user to blink [48]. Furthermore, Xu et al. [12] created a system that can use a person's public photos to construct realistic, textured, 3D facial models that can bypass systems that include a liveness detector[3] or are motion-based. Another problem with fingerprint and face recognition is that they are available to be captured by anyone, and once a fingerprint or a face photo is stolen it can not be recovered. When users suspect that their password was stolen, they can simply change it. But if a fingerprint or a face photo is stolen, it can not be changed. In addition to security, these security methods hinder usability of devices. For example, users find password entry on mobile devices more annoying than lack of coverage, small screen size, or poor voice quality according to the research done by Shi et al. [10]. Furthermore, users tend to use insecure passwords because they do not believe that their assets are at risk and lack the motivation to make extra effort [46]. As for the usage of additional security hardware, a study showed that users are not receptive to the use of security tokens [17].

1.2.2 Why Mobile Devices?

An important aspect of contextual authentication is that it can be used on any device which provides data that can be used to identify the user. An example of these kinds of devices is a mobile phone. Mobile phones have many characteristics that make them great candidates to be main instruments for contextual authentication. Mobile devices usually have one owner, which means that there is a relation between the owner and his/her device. This relation can be exploited for authentication purposes. For example, each person has a unique way

[3]Liveness detection is to determine if the biometric being captured is an actual measurement from the authorized, live person who is present at the time of capture [39]

to hold and pick up the phone [10], types on the keyboard, scrolls, and flings the screen in patterns that can be differentiated from other people [2]. Even the gait of a person can be captured by the phone and used to identify the user [15]. Another reason is the continuous evolution of mobile phones over the last decade. They are integral elements of people's lives because of their capabilities, and as such they contain a vast amount of private information which require protection, especially because passwords, PINs, and patterns can be easily stolen via shoulder surfing attacks and smudge attacks. Finally, mobile devices have one more important characteristic that makes them lucrative for this kind of research: they have an incredible amount of rich data sources that can be taken advantage of. This rich data includes:

- Location: The location of mobile device can be acquired by different methods: GPS, IP address, and the signal strength of any wireless network nearby.
- Sensors: Mobile phones come equipped with variety of sensors that can measure phone acceleration, ambient light, and proximity. New devices can even measure ambient temperature, air pressure, and humidity [53]. The accelerometer and orientation sensors can be used to identify patterns such as the gait of person and the way the phone is picked up.
- Battery usage can be used to track and identify users [59].
- Application usage and file system changes.
- Logs, such as calls log, can be used to compute a pattern and check for abnormalities.
- A touch screen can provide lots of useful data that can be used to identify users. The various actions that a user performs when interacting with the phone through the touch screen (tap, fling, and scroll) can be used to recognize the user.
- Network activity: users usually spend their time in regular places where their phones are connected to a familiar cellular tower or wireless hot-spot. This can be used to identify abnormal behavior. For example, a user might be denied access to his bank application if he is not connected to the "home" wireless network.
- USB connections: connecting a phone to a computer that was never connected to before might raise suspicion. This can be used as a factor to ask for explicit authentication.

From what has been introduced thus far, We can conclude that: (1) using implicit authentication can serve the users' needs for more security and also more usability, and (2) mobile phones can serve as the ideal medium to apply contextual authentication.

1.3 Contribution

The focus of this work is to create an authentication mechanism that takes advantage of the rich data that a mobile phone produces. This authentication tool will not require the user to make any additional effort beside using the phone. It will focus on using the natural movement of the phone for the authentication purpose. Natural phone movements are those which happen when the user is actually using his/her phone, such as typing. This means that the user will not be required to perform any additional movements or tasks for the authentication mechanism to function, the tool will exploit the regular way phone owners usually use their devices. The users, in this case, will be authenticated to their phones implicitly. As a result, this provides the potential to increase both security and usability. To create such a mechanism, specific motions (that is natural to phone usage) must be selected, data needs to be collected from participants while performing these motions, features need

to be extracted from this data, and machine learning needs to be utilized. These activities will be explained in greater depth during the following chapters.

The rest of this thesis is organized as follows. In Chapter 2, we review the work done in the field of contextual authentication including movement based context, touch screen based context, and other frameworks. We define the research questions in Chapter 3. Then, we present in Chapter 4 the essential knowledge required to understand this thesis. In Chapter 5, we illustrate the implementation of a proof of concept for a context-based authentication method. The results of the implementation are then presented in Chapter 6. Finally, Chapter 7 will include the answers to the research questions, future work, and conclusions.

The research on contextual authentication in mobile devices can be categorized based on the context used to identify users. These are: movement based, touch screen based, other types, and general frameworks.

2.1 Movement Based Context

The availability of acceleration sensors in mass-marketed communication devices creates exciting opportunities for data mining and data mining applications [1]. The first of these applications was activity recognition. An example of this is the work done by Su et al. [21]. They built a system to identify user activities such as walking and jogging using data form acceleration sensors and were able to achieve an overall accuracy of 91.7%.

Taking advantage of the acceleration sensors a step further for authentication purposes, Derawi et al. [15] created a prototype using a commercially available mobile smartphone. The prototype identify users based on their gait. Gait recognition is considered user friendly because the gait of a person can be captured unobtrusively and continuously, unlike for example a person's fingerprint or retina. It also considered secure due to the fact that the gait of an individual is difficult to mimic [44]. However, there are challenges when identifying users using gait recognition. These challenges are because gait will be affected by (1) stimulants, like drugs and alcohol, (2) physical changes, for example pregnancy, an accident or disease affecting a leg or foot, or severe weight gain/loss, (3) psychological changes, where the mood of a person influences his/her gait, and (4) clothing and in particular shoe wear. In their experiment, 25 volunteers were asked to walk for approximately 30 meters for three different walking speeds (slow, normal, and fast). Five of these volunteers' data was used as templates, while the remaining 20 volunteers' data was used for testing. For classification, the Manhattan distance metric was used on the phone, and it resulted in 28% false negative and 19.5% false positive. The data was analyzed later on a computer where they used the cosine distance metric which had better results (10.7% false negative and 1.4% false positive).

Conti et al. [4] argued that walking is not naturally connected to the phone usage. Thus, they suggested another method to identify users using accelerometers as the data source. They proposed that smartphone users can be identified by the way they answer or place a call. They argued that the call answering movement is a unique biometric feature. They proposed the use of accelerometer sensor (S_a) and orientation sensor (S_o). As for classification, they implemented two extensions for the Dynamic Time Wrapping (DTW) algorithm; (Dynamic

Time Warping Distance (DTW-D), and Dynamic Time Warping Similarity (DTW-S)). For testing, they created an Android application that logs the values sensed by the accelerometer sensor and the orientation sensors. They asked 10 users to use the application to trace data of several movement patterns. They decided to try three methods to check the performance of their system. First, they took the binary output of each algorithm with each sensor (i.e., DTW-D with S_a, DTW-D with S_o, DTW-S with S_a, DTW-S with S_o). The binary output is simply *0* if the user is rejected by the system, or *1* if accepted. In the best case, they were able to achieve a 4.4% FAR[1] and 9.3% FRR[2] using DTW-D with S_o. Second, they tried several combinations of the methods mentioned previously. As an example, the system may accept a user only if n of the basic methods accepted him/her (n = 1, 2, 3, or 4). In the case of n = 1, the FRR can improve to less then 1% meaning that the system rarely rejects the right user. However, the FAR becomes higher than 40% meaning that the system accepts too many intruders. The reason for that is because using n = 1 means that the system does not have strict threshold, or more simply put, a user needs only to pass 1 threshold to be accepted by the system. If n was chosen as 4, the exact opposite happens. Finally, they tried a combination of normalized versions of the (DTW-D) and (DTW-S). Using these normalized versions they were able to improve the FRR from 9.3% to 8% and the FAR from 4.4% to 2.5%.

Another example is the work done by Chang et al.in [3]. In their experiment, they embedded accelerometers in TV remote controls and showed that users can be distinguished based on the way they use the remote control. They used many features in their study, and achieved an accuracy ranging from 70 to 92% depending on the features used. The one worth mentioning is the usage of 3-axis acceleration parameters. Their observation regarding hand motion pattern indicates that users might be distinguishable by looking at acceleration features when the controller is picked up or put back down.

2.2 Touch Screen Based Context

There has been a lot of research on creating authentication systems based on phones' touch screens. The reasons for this are: (1) the touch screen is the main input method for phones, and (2) the amount of features that can be extracted from interacting with the touch screens are plenty. These features include pressure, velocity, size, speed, duration, and area.

Seo et al.[26] developed a biometric identification method for mobile devices by analyzing the user's input patterns, such as a finger's touch duration, pressure level and the touching width of the finger on the touch screen. For training, they used a Back Propagation Neural network (BPN), which uses a supervised learning method. Feed-forward architecture was used as the pattern classification algorithm. Training was done on a desktop computer because a specific level of calculation is required to learn a user's input pattern. Their experiment included 50 participants aged 20 to 30 years old. Each Test participant took a touch and scroll-wheel input test ten times. They collected seven kinds of input pattern data: input duration, the pressure level and the touching width of the finger, scroll-wheel's start and finish positions (x and y coordinates), speed, and length. They claimed that their neural network can identify the real user with a 99.6% success rate after 4 scroll gestures.

A year later, Bo et al. [2] created an android app called SilentSense. Similarly, their system exploits the user touch behavior biometrics and leverages the integrated accelerometer and touch screen sensors to capture the micro-movement of the device caused by user's

[1] False Acceptance Rate, the measure of the likelihood that the biometric security system will incorrectly accept an access attempt by an unauthorized user [47].

[2] False Rejection Rate, the measure of the likelihood that the biometric security system will incorrectly reject an access attempt by an authorized user [47].

screen-touch actions (tap, fling, scroll). In addition, they added a movement-based biometrics for each user with previous touch-based biometrics. FAR of identification by one observation of tap is 22%, by one fling action is 9%, by one scroll action is 23%. With about 12 observations of various actions, the FAR and FRR are both reduced to nearly 0, meaning that there is no incorrect identification. In a dynamic situation (user is walking), the accuracy to identify the owner can achieve 100% after 7 steps.

In the previous examples, the authentication operation occurs during the actual usage of the phone after the login stage. Angulo et al. [40] developed an Android application where the authentication procedures takes place at the login stage. The application uses lock pattern dynamics as a secure and user-friendly two-factor authentication method. Their application checks the way a user draws the unlock pattern on phone's screen. For training, they asked 32 persons to draw three different patterns 50 times each. They disregarded the first 10 trials as they considered the participants will use the first few tries for practice. Then, they used the next 25 trials to train the classifier. The remaining 15 trails were used for testing. They used Random Forest as the learning method for the classifier. Two main features were captured for each successful trial: the *finger-in-dot* time, which is the time in milliseconds from the moment the participant's finger touches a dot to the moment the finger is dragged outside the dot area, and the *finger-in-between-dots* time, representing the speed at which the finger moves from one dot to the next. With this setup, they were able to achieve an FAR of 5% and FRR of 14.16%.

In [5], Feng et al. created an authentication system for mobile devices called TAP that combines the ideas of the two previous examples. TAP leverages the biometric information embedded in the typing habit and hand morphology to identify users and attacker during login stage (using password) and post-login stage. The key typing features they included are time interval between two character, press time of a character, and pressure while pressing a character. They were able to achieve a FAR of 10.4% and a FRR of 11.1% in login stage and a result of 8.3% FAR and 5.6% FRR during post-login stage for 20 character long input sequences.

Li et al. [8] proposed a biometric-based system to achieve continuous and unobservable re-authentication for smartphones which is similar to the work of Bo et al. [2]. The system uses a classifier to learn the owner's finger movement patterns and checks the current user's finger movement patterns against the owner's. The re-authentication module is deployed in a smartphone and the training module is executed on a computer. For training, 28 people were asked to use smartphones till at least 150 sliding up gestures, 150 sliding down gestures, 150 sliding right gestures, 150 sliding left gestures, and 300 tap gestures were collected. For testing, 47 people were asked to use the phone until the total using time hit 15 minutes. Using Support Vector Machine for training, and 14 sliding gestures for authenticating, they achieved an EER of approximately 4%. Using less than 14 sliding gestures increases both FAR and FRR.

In 2013, a year after Angulo et al. [40] created their authentication application, Shahzad et al. [9] proposed a similar scheme called GEAT. This scheme authenticates users based on the way they unlock their phones through a pattern on the touch screen. The scheme first asks the user to perform gestures (drawing simple patterns) on the screen for 15-25 times to obtain training samples. Seven features are extracted from these samples to build a reference set. These features are: velocity magnitude, device acceleration, stroke time, inter-stroke time, stroke displacement magnitude, stroke displacement direction, and velocity direction. Then, when a user unlocks his or her phone, the gesture is captured and features are extracted and compared to the reference set. Their results showed that GEAT can achieve a better Equal Error Rate (EER) than their predecessor. The achieved EER is 0.5%

with 3 gestures (swiping or pinching) using only 25 training samples.

Touch-based authentication might sound bullet-proof. However, Serwadda et al. [27] were able to design a Lego robot which can perform attacks on touch-based authentication systems. They presented two types of attacks: (1) a population statistics-driven attack in which patterns gleaned from a large database are used to formulate the input supplied to the robot, and (2) a user-tailored attack which uses samples stolen from the victim in question as a basis to formulate input to the robot. They argued that this kind of attack is easy to launch. The Lego robot used to implement the attack can be easily operated by a middle school student [22], whereas the population data required to launch the population-based version of the attack is easily accessible on the Web. Also, the robot just has to swipe on the phone screen with no knowledge of the design of the authentication software. They showed that the population-based attack increases the mean false acceptance rate (FAR) of the majority of the classification algorithms by more than 100% of the mean FAR seen before the attack. The user-specific attack even has a much more drastic effect on the authentication system, driving the classifier FAR to almost 100%.

2.3 Other Types of Context

While most of the implicit-based authentication work was done using movement or the touch screen features as the context, other researches approached the problem from a different perspective.

Shi et al. [10] created a prototype for android phones where they used the following features:

- Frequency of good events[3] (elapsed time since last good call/message/browse).
- Frequency of bad events (number of times per day for bad call/message/browse).
- GPS coordinates.

They used a set of 50 users to create models for legitimate users, which characterize the user's behavioral patterns. These characteristics include, for example, how frequently the user typically makes phone calls to numbers in the phone book. They concluded that an adversary who uses the phone as his/her own after stealing will be detected after using the device 2 times, 3-6 times, 7-10 times, 11-16 times with probability 50%, 75%, 90%, and 95% respectively.

Witte et al. [11] proposed a context-aware mobile system to identify users. They used the following aspects as the context factors: accelerometer, magnetic field, screen brightness, battery usage, screen state (on/off), call duration and direction, and noise. They categorized these features by the type of sensor (physical such as the accelerometer and logical such as the battery) and by the way the features is collected (active and passive: active is when the sensor can be activated and data can be collected in predefined intervals and passive is when data is gathered based on user interaction). For training, they collected data from 15 participants and used SVM algorithm for model creation. To evaluate their model, they calculated the precision P and recall R using these equations:

$$P = \frac{T_p}{T_p+F_p} \qquad R = \frac{T_p}{T_p+F_n}$$

[3] A good (bad) call/message is a call/message (was not) made to or received from a number on the contact list. A good (bad) browse is visiting a link which was (not) visited before.

where T_p is true positives, F_p is false positives, and F_n is false negatives. The precision rates were between 0.74 and 0.92 while recall rates were between 0.15 and 0.86. They concluded that some features such as battery usage is not useful at all while other features such as call duration and brightness represent the most reliable features.

Kayacik et al. [20] designed a context-based authentication system that is temporally and spatially aware. Their solution compares current behavior with a user profile. A detection threshold is computed based on the user's security settings to activate explicit authentication when the comfort (device's confidence in the identity of the user) is below the threshold. The application has two modes: training and deployment. In training mode the application builds a profile for the user and establishes a threshold. In deployment mode the application monitors sensors data and trigger an explicit authentication request when comfort goes below the created threshold. The main idea about their solution is using time and location from cell towers as anchors. Each anchor focuses on a specific location or time and describes the general characteristics for that anchor through a set of probability density functions for each sensor's data. For example, when the user is at home, he/she might not access work email. In the temporal model there are 24 anchors, one for each hour of the day. On the other hand, the spatial model may contain as many locations as the user visits. As for the number of sensors used, it varies per anchor. To evaluate the system, they asked a participant to use the phone in training mode to build a profile. Then, when in deployment mode they devised 4 attacks: uninformed outsider, informed outsider, uninformed insider, informed insider. An uninformed attacker does not know how the system works while an informed attacker does. An outsider has no info about the usage patterns of the user while an insider does. The detection rate for the attack scenarios were 99.44%, 98.12%, 95.85%, and 53.21% respectively.

2.4 Frameworks

In addition to the previous implementations, there are other schemes that instead of providing the actual authentication procedures they introduce frameworks. These frameworks provide a groundwork which allows other developers to use it as the foundation for their applications and research into contextual authentication.

Crawford et al. [14] developed a framework to provide continuous and transparent mobile device authentication. In their work they did not develop any of the context-based authentication mechanisms, they just built the system that combines and processes data from other implementations. Their framework uses multiple behavioral biometrics which, together, deliver a device confidence level C. Each task on the device has a threshold C_T which can have a default value or can be set by the user. When the task is requested, the device confidence C is checked. The task is then approved if C_T is lower than C. If it is higher, then an explicit authentication is required. To test their concept, they ran a simulation in which they modeled an owner who executes a random task at random intervals, and an attacker who obtains the device in an unlocked status. For authentication, they used keystrokes dynamics and voice verification as implicit methods, and PIN as explicit method. The tests showed that the owner will only need to authenticate himself explicitly 73 times in 24 hours period, while on the other hand he needs to authenticate himself 218 times if only PIN authentication was used. The attacker, in this model, was only able to perform one task before the device detected a low confidence and requested explicit authentication.

In [7], Khan et el. developed a framework for Android called Itus. This framework allows app developers to use context data to provide implicit authentication by changing as few as two lines of their existing code. Moreover, Itus provides an oracle capable of making

advanced recommendations, such as optimal sets of behavioral features and classifiers that can be used in particular apps, should developers wish to fine-tune the classifiers. This framework aims to serve: (1) app developers who only want to add support for implicit authentication to their apps without tuning its accuracy, and (2) app developers who wish to fine-tune Itus for accuracy and performance. In the first case, app developers only need to identity activities in their apps that should be protected and extend them using Itus library. In the second case, app developers can use the Itus oracle on a desktop which can be fed with new data which in turn gives recommendation.

2.5 Conclusions

Researchers have used touch screen or movements sensors as a method to create context-based authentication mechanisms. Although they were able to achieve good results, each method (touch-bases and movement-based) has its own problems. Touch-based authentication systems can be attacked by a Lego robot which can tremendously diminish their accuracy [27]. Gait-based authentication system achieved in [15] has a main drawback. This system depends on a movement that does not happen when the phone is being used, i.e, not natural to phone usage. Moreover, there are many external factors, such as the type of shoe worn, that can negatively affect the accuracy of this system. Finally, the work done in [4] accomplished good results using call placement or call answer movement as the authentication method. The problem with this method is that it requires a specific movement to occur in order for the authentication mechanism to kick in. This would leave the phone vulnerable in other situations.

With that being said, it is undeniable that context-based authentication is a promising field and it demands more research in order to increase the confidence of security specialists and users in such systems.

As explained in Chapter 1, classical security mechanisms are not reliable enough. For example, users tend to use insecure PINs and passwords. Also, these PINs and passwords are susceptible to shoulder surfing attacks and smudge attacks. Moreover, security specialists no longer trust the password to do its job and companies foresee that they will do away with passwords in 5-9 years. Other types of security mechanisms, such as face recognition and finger print, provide more security. However, even these types of advanced security are vulnerable to hacking. In addition, they hinder the usability of the mobile device. The solution for these problems is by using context as an authentication method. The reason behind this is because (1) context improves security by utilizing human characteristics and the relation between phones and their users, and (2) it does so without requiring the user to make extra effort and thus improve usability.

Researchers have already developed several prototypes that use context in mobile phones to authenticate users. Most of these prototypes use touch screens as the context factor, utilizing the gestures that a user perform on the screen to authenticate the user. However, we have seen that Serwadda et al. created a Lego robot that can perform attacks on these kinds of authentication systems and significantly diminishes their accuracy [27]. To solve this problem, movement-based authentication can be used. For example, researchers used the gait [15], and call answering or call placement [4] as the context factor. That being said, both these movement-based authentication techniques have problems. First, the gait is not a phone natural movement, and gait can be affected by external factors which will reduce the accuracy of the system. Second, in the call answering/placement method, the problem resides in the fact that the user need to answer or place a call for the authentication mechanism to work. A possible solution for these problems is an authentication mechanism that does not require the user to perform specific movement with the phone. This kind of system should use a movement that is: (1) natural to the phone usage, and (2) happens frequently when the phone is used.

3.1 Research Questions

We formulate here the main question of the thesis followed by the sub questions.

3.1.1 Main Question

Finding an answer to the main question serves as the goal of this work. The discussion at the beginning of this chapter leads us to formulate the main question of the thesis.

Main question: Can we provide continuous and implicit authentication from owner to mobile device with high accuracy while utilizing mobile devices' natural movements?

Since activities on mobile devices vary in sensitivity, and since users usually use weak PINs or password for the sake of usability, a system that continuously authenticates users in the background can increase security by using multiple sources to identify owners and also increase usability by reducing the amount of times users need to explicitly authenticate themselves. With more specific training, special cases can be created that can serve the user in situational circumstances, such as training the phone to recognize the way it is picked up from its owner pocket or purse.

3.1.2 Sub Questions

The sub questions are important tools to help answer the main question of the thesis. They can serve as milestones on the road to answer the main question.

Sub questions:

1. Can an owner of a mobile device be distinguished from other users by recognizing: (1) the way the owner picks up the phone, and (2) micro-movements of the phone during interactions?

By answering this question we can reach a conclusion whether these two specific motions have enough characteristics that can be used as a unique identifier for mobile users, and thus serve as a security mechanism. It is important to mention here that the two motions (off-table pick-up and micro-movements) satisfy two conditions mentioned in the main question which are "natural to phone movement" and "implicit". The micro-movement motion also satisfies the condition which is "continuous". Another important factor about these two movements is that the pick-up motion usually precedes any other other movements, meaning that a user would need to pick up the phone before using it. Hence the pick-up motion serves as the first line of defense. On the other hand, the micro-movements happen continuously while the user is using the phone and usually after a pick-up motion had occurred. Thus, this motion can serve as a second, and continuous, line of defense.

To answer this question, sensor data will be collected from different participants and then analyzed. The aim of this analysis is to check if the suggested movements are unique to the owner of the mobile device. After that, machine learning techniques will be used to build a classification model for the owner and tested against other participants.

2. What kind of features can be extracted from collected sensors' data during the previously mentioned movements that have high discriminating characteristics that can be utilized to recognize the owner?

It is a critical part of machine learning to find features in the data in question that can truly give a distinction for this data. Without these features, collected data will not provide any value that can be used to achieve the goal, which is being able to separate owner data from non-owner data. These features can range from very simple such as the average to more complex such as energy. However, using all the extracted features is usually not the best thing to do, and selecting a number of features (feature selection) from all the extracted ones need to be done. The reasons for feature selection are: (1) some features might be irrelevant to the data and model being built, (2) some features might be redundant, (3) fewer features increase the performance of training and prediction, and (4) feature selection

counters overfitting[1] by doing more generalization. That being said, these kinds of features are not needed sometimes. This is because by calculating these features, other important characteristics might be lost. For example, if a stream of collected data is time related then calculating the average might not be a good feature that describe the data. In such a case, algorithms that works better with time-series can be used.

3. What are the appropriate classifiers and parameters which work best with these kind of movements?

There are many types of classifiers that can be used. Choosing the right one requires an understanding of the data and sometimes might require trial and error. For the pick-up motion, since this movement can be considered as a time-series, a classifier that works with time-series needs to be chosen. Another factor that plays a role in choosing the right classifier is the number of classes that the model need to support. In our case, two classes are present: owner and non-owner. An additional factor is the number of training samples. Some classifiers need huge amounts of training samples to perform well, while others do not. In our case, the intent is to try to accomplish good classification accuracy without the need for lots of training samples as to create the authentication model without the need for the application to spend too much time collecting data from the owner.

In addition to the algorithms themselves, different options and parameters for each algorithm need to be chosen and tuned. Those options and parameters are important and need to be chosen carefully in order for the model to truly represent the problem to be solved, and to optimize the algorithm's performance. Some options are chosen based on the definition of the problem and the kind of prediction the model ought to make. For example, choosing a classification or regression model depends on whether the model needs to predict new value for the new (unseen) input or needs to predict to which predetermine group (class) the new input belongs. Other parameters can be chosen through the process of hyperparameter optimization[2]. Examples of hyperparameter optimization are Grid Search, Bayesian Optimization, Random Search, and Gradient-Based Optimization.

4. How to correctly detect the pick-up motion at the right time so that the right data can then be sent to the authentication process?

It is very important to detect the pick-up motion at the right time and fast enough. This is because: (1) if the motion is not detected the authentication mechanism will be useless, (2) if non-pick-up motions were detected, the wrong data will be sent to authentication process which will lead to a wrong decision, and (3) if not detected fast enough it could hinder usability, or it could lead to a late decision that might get exploited.

There are two possible approaches that can be experimented to detect the pick-up motion. The first one is by analyzing the pick-up motion data from all participants and find a set of thresholds that the motion pass or not pass and create set of rules based on these thresholds. This could have the advantage of fast performance. The second approach is using machine learning to build a model that describe the pick-up motion in general using the data of all participants. This might be slower than the first approach and will lead to at least two prediction calculations (one to detect the motion and one to make the decision).

[1] Overfitting occurs when the model represents noises and errors rather than the real characteristics of the data.

[2] The process of finding set of parameters for a learning algorithm to optimize the performance.

This chapter gives an overview of some of the technologies and foundational knowledge required to understand this thesis. It gives some facts on the sensors available in Android phones and their different capabilities. It also details machine learning; what it is, different types of machine learning, and different algorithms.

4.1 Development Platform

The platform of choice in the project is Android. There are two main reasons for choosing Android over iOS: (1) Android has a market share of 82.8% while iOS's market share is 13.9% [55], and (2) deploying an app onto an Android device is free and easy and can be done through the Android Studio[1] developed by Google. In contrast, to be able to deploy an app onto an iOS device, a fee of $99 a year must be paid to Apple [2].

4.2 Sensors Overview

Most Android-powered devices have built-in sensors that measure motion, orientation, and various environmental conditions. These sensors are capable of providing raw data with high precision and accuracy, and are useful to monitor three-dimensional device movement [53].

The Android platform supports three broad categories of sensors:

- Motion Sensors

  These sensors measure acceleration forces and rotational forces along three axes. This category includes accelerometers, gravity sensors, gyroscopes, and rotational vector sensors. Motion sensors are useful for monitoring device movement, such as tilt, shake, rotation, or swing. The movement is usually a reflection of direct user input, but it can also be a reflection of the physical environment in which the device is sitting [51].

- Environmental Sensors

  These sensors measure various environmental parameters, such as ambient air temperature and pressure, illumination, and humidity. Environmental sensors supported on the Android platform include barometers, photometers, and thermometers. Those

[1] https://developer.android.com/studio/index.html
[2] https://developer.apple.com/support/compare-memberships/

environmental sensors however are not standard sensors and can only be found in specific models.

- Position Sensors

  These sensors measure the physical position of a device. This category includes orientation sensors and magnetometers. Figure 4.2 shows the three orientation angles of a mobile phone.

In this project, acceleration sensor and orientation sensor are going to be used to collect data which in turn is going to be used for training and testing. Following is an overview of those two sensors.

4.2.1 Acceleration Sensor

An acceleration sensor (accelerometer) is an electro-mechanical device that measures acceleration[3] forces. These forces may be static, like the constant force of gravity, or they could be dynamic, caused by moving or vibrating the accelerometer [50]. In android, the accelerometer is hardware-based and when queried it returns multi-dimensional array of sensor values for each SensorEvent[4]. For example, during a single sensor event the accelerometer returns acceleration force data for the three coordinate axes.

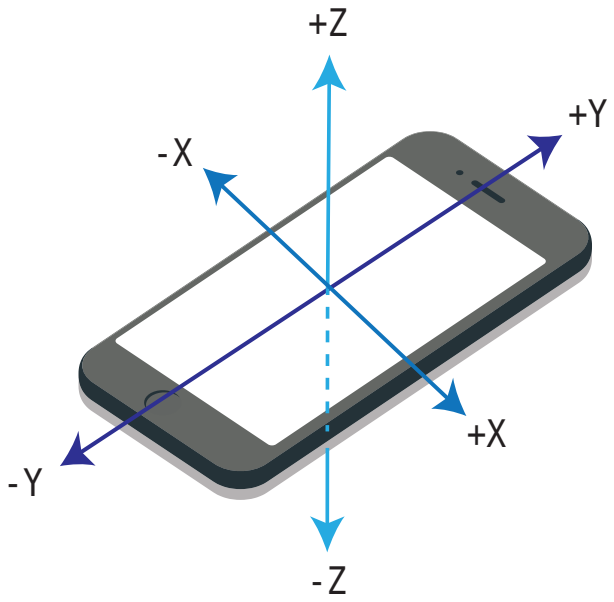


Figure 4.1: Acceleration along the X, Y, and Z axes

4.2.2 Orientation Sensor

The orientation sensor was deprecated in Android 2.2, so in order to obtain the changes in device orientation, magnetometer[5] and accelerometer need to be used together to acquire

[3] Acceleration is the measurement of the change in velocity, or speed divided by time.

[4] SensorEvent is a class in Android represents a Sensor event and holds information such as the sensor's type, the time-stamp, accuracy and the sensor's data[52].

[5] A sensor that measures the strength of the magnetic field.

the required data. This can be accomplished by invoking the getRotationMatrix[6] method in Android which will return the azimut, pitch, and roll as shown in Figure 4.2.

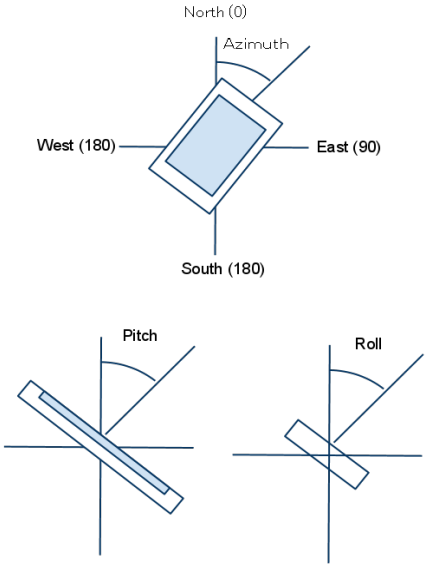


FIGURE 4.2: Orientation angles with regard to the north pole[7]

4.3 Machine Learning

Machine learning is a crucial part to any kind of context-based authentication. This is due to the fact that the device which the user is trying to authenticate to needs to learn the usage patterns of its users in order to recognize them later on.

A simple yet comprehensive definition of machine learning was giving by Arthur Samuel as: "Field of study that gives computers the ability to learn without being explicitly programmed" [37]. In more complex terms, machine learning is programming computers to optimize a performance criterion using example data or past experience using a defined model of parameters [32]. Machine learning can be classified into supervised learning and unsupervised learning. Supervised learning is when there is an input, X, an output, Y, and the task is to learn the mapping from the input to the output [32]. In unsupervised learning, there is only input data and the aim is to find the regularities and structure in the input.

4.3.1 Classification & Regression

Supervised learning by itself can be categorized into two sets: **Classification** and **Regression** [32]. Classification is the problem of determining to which category a new set of data (observation) belongs. This is done using a training data set whose category is known. In the training phase, each input set belongs to a specific category, and when new input arrives, the purpose of the system is to determine to which one of the output categories the input belongs. For example, the input data can be the speed of cars on a road and the output is whether the car crashed or not.

[6] https://developer.android.com/reference/android/hardware/SensorManager.html
[7] http://d.hatena.ne.jp/IchiRoku/20101219/1292731795

Regression is a statistical process for estimating the relationships among variables [63]. For example, the training data can be the size of houses as input and the price of houses as output. For prediction, the system needs to estimate the price of houses of any size.

4.3.2 Algorithms

Classification and regression algorithms can be used for time-dependent data, also known as time-series, and time-independent data.

Time-series are ordered sequence of values that have been collected over fixed sampling intervals [33, 57]. In contrast, time-independent data are observations collected at the same point of time, or without regard to differences in time.

The most common methods for time-series data classification are Hidden Markov Models (HMM), Dynamic Time Warping (DTW), Recurrent Neural Networks (RNN), and Dynamic Bayes Nets (DBN). In practice, HMM and DTW are the most used methods for this kind of work. The authors of [42] performed a thorough comparison between DTW and HMM. According to their results, DTW performed 2% better than HMM in gesture recognition. Moreover, they showed that HMM needs more training data to reach similar results with DTW. As a result, DTW will be the algorithm of choice for time-series data.

For time-independent data classification and regression, the most used algorithms for gesture recognition are Random Forest, Support Vector Machine, Naive Bayes, and Logistic Regression. Support vector machine works better than other algorithms when: (1) training samples are few, and (2) 2-3 classes are being used. We will see later that this suits our needs and so it will be the chosen algorithm for time-independent data.

4.3.2.1 Dynamic Time Warping

Dynamic time warping (DTW) is a well-known technique to find an optimal alignment between two given (time-dependent) sequences under certain restrictions [35]. It provides both a distance measure that is insensitive to local compression and stretches and the warping which optimally deforms one of the two input series onto the other [18]. DTW has been used for classification and clustering in diverse domains: on-line signature recognition [16], fault detection in bioprocess applications [19], aligning gene expression [13], electrocardiogram (ECG) frame classification [6], and much more.

The DTW algorithm works by trying to find the optimal alignment between two time series when some of these two series had been warped in time (stretched or shrunk). This warping between the two time series is used to find the similarity between them. Figure 4.3 shows two times series warped to each other. Each vertical line connects one point of a time series to a point that it is similar to it in the other one. This is what warping is. If the two time series were identical then the vertical lines would be perfectly vertical. The goal of DTW is to find the warp distance which is the measure of the difference between the two time series. This warp distance can be calculated by summing the distance between each pair of points connected with a vertical line. So if two time series are identical the warp distance would be zero.

Mathematically, DTW can be explained as follows:
The two time series in Figure 4.3 can be described as:

$$X = x_1, x_2, ..., x_M$$
$$Y = y_1, y_2, ..., y_N$$

A warp path is constructed as follows:

$$W = w_1, w_2, ..., w_K$$

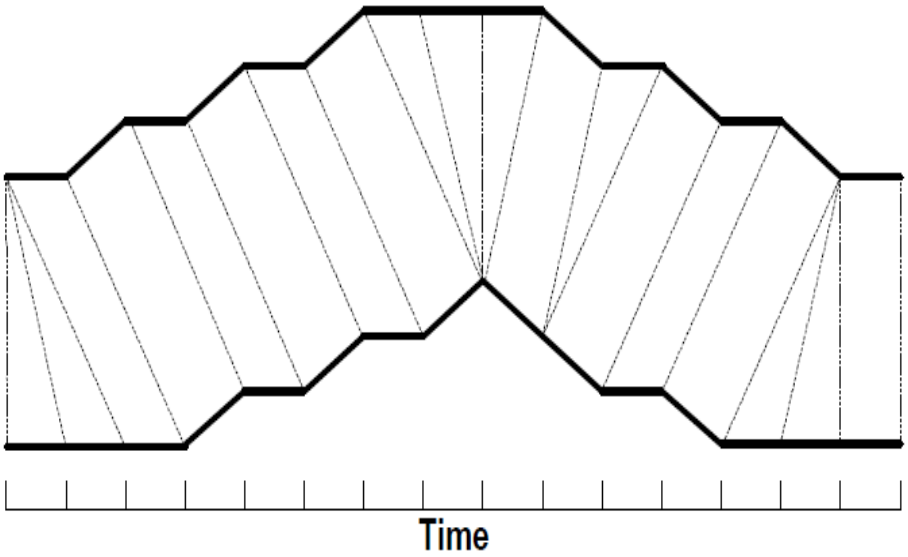


FIGURE 4.3: Two Warped Time Series [24]

where K is the length of the warp path and $w_k = (x_m, y_n)$. The warp path must start at the beginning of both time series $w_1 = (1, 1)$ and must finish at the end of them $w_K = (M, N)$. This to ensure that every point of the time series is used. Finally, the optimal warp path needs to be found, which is the warp path with the minimum distance. The distance of a warp path is calculated as follows:

$$Dist(W) = \sum_{k=1}^{k=K} Dist(w_{km}, w_{kj})$$

where Dist(W) is the distance of a warp path calculated using Manhattan distance or Euclidean distance, and $Dist(w_{km}, w_{kj})$ is the distance between two data points from series X and Y in the k^th element of the warp path.

4.3.2.2 Support Vector Machine

Support Vector Machine (SVM) is a training algorithm that maximizes the margin between the training patterns and the decision boundary. A key characteristic of SVM is that it concentrates on classes' points that are considered difficult to differentiate from each other, while other algorithms concentrate on all the points. The rational behind this is that if a classifier is good at separating classes at those difficult points, then it performs even better when dealing with easier points. For example, when searching for a plane that can separate classes A and B in Figure 4.4, the best plane is the one that can maximize the distance between points of B closest to A and vise-versa. So in this case the vertical plane is better than the diagonal one. Again, it is not important to take into consideration all the points of the classes to accomplish this separation. Another aspect is that SVM, dissimilar to other algorithms, is explicitly instructed to find the best separating plane. It does this by performing the following steps: (1) it finds the closest points, (2) it draws a line between them, and (3) it creates the separating plane which is the line bisecting the connecting line and is perpendicular to it. This is shown in Figure 4.5. When new points are introduced, the algorithm had already found the best separating line that keeps A and B as far away from each other as possible which will lower the chance that new points will overflow from

A to B or from B to A. It is important to notice that the previous example is dealing with linearly separable data. In case the data is linearly non-separable, other types of planes can be used. These planes are also known as kernels. For example, a linear kernel is the same as the one described before. Another example is radial based function, in which the separating plane would be radial. Additional types include Polynomial and Sigmoid.

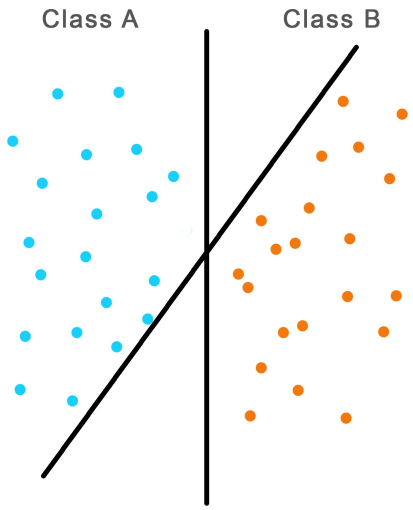


Figure 4.4: The vertical plane is better in separating the two classes because it maximizes the distance between points of B closest to A and vise-versa

SVM Parameters:

There are four important parameters to consider:

- Type: The type of the SVM such as classification or regression. In this case, we are using the input data to predict whether the user is the owner or not. This can be translated into two classes: the owner being the first class and other people being the second class. As a result, classification is the type to be chosen.
- Kernel: Kernels are a method for pattern analysis which deals with the automatic detection of patterns in data. The available kernels include: Linear, Polynomial, Radial Base Function, and Sigmoid. Linear kernel usually used when the different training data can be linearly separated. If the training data is not linearly separable then one of the other kernels must be used.
- Gamma: defines how far the influence of a single training example reaches. This is usually chosen through a grid search.
- C: trades off misclassification of training examples against simplicity of the decision surface. This is usually chosen also through a grid search.

4.3.3 Fast Fourier Transform

Fast Fourier transform is an algorithm which computes the discrete Fourier transform of sequence of data. A process can either be described as a function in time domain or in the

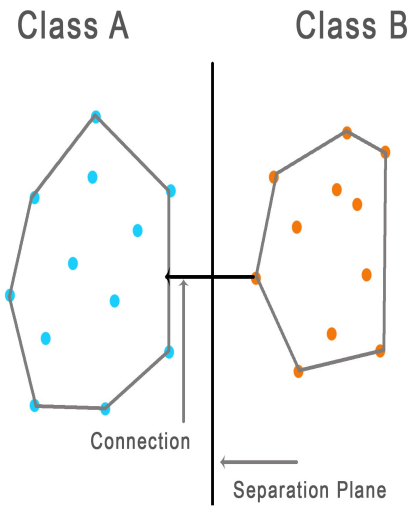


FIGURE 4.5: Determining the best plane by: (1) finding the closest points, (2) drawing a line between them, and (3) creating the separating plane

frequency domain [36]. These two descriptions can be considered as two representations for the same function. The Fourier transform of a function of time itself is a complex-valued function of frequency, whose absolute value represents the amount of that frequency present in the original function, and whose complex argument is the phase offset of the basic sinusoid in that frequency. It basically allows a signal to be viewed in different domain and hence provides more options to study and analyze the signal.

A Movement-Based Authentication Mechanism 5

This chapter will demonstrate the steps we have taken to implement movement-based authentication mechanisms. One of the mechanism will be based on the pick-up motion, and the other on micro-movements. The implementation will include data collection, training, and testing. The chapter will also explain the creation of a pattern recognition module for the purpose of detecting pick-up motions.

The implementation begins with developing a simple android app to collect sensors' data while participants are performing a pick-up motion or typing for micro-movements. This data includes the acceleration on three axes (X, Y, and Z) and the orientation angles (Yaw, Pitch, and Roll). The collected data will be used in the second part of the implementation. The second part focuses on training and creating a classification model. Here, a subset of the collected data in the first part will be used for training, while the remaining subset will be used for testing. As for the algorithms, DTW will be used for the pick-up motion and SVM for the micro-movements. The reason for these two choices is explained in sections 5.4.1 and 5.5.1. After training is finished, part three takes place. In the third part, the classification models created in previous part will be tested. We will use the remaining collected data which was not used in the previous part. The goal of this part is to determine the accuracy of the classification models, mainly FAR and FRR. In the fourth and final part, we will implement a pattern recognition module. The purpose of this module is to correctly detect an occurrence of a pick-up motion in order to eventually send the data to the authentication module and receive a decision.

5.1 Device Specifications

To collect the essential data and test the developed apps, a Samsung Galaxy S Duos was used which has the following specifications:

- Android OS, v4.0.3 (Ice Cream Sandwich).
- Magnetometer, Proximity sensor, and Accelerometer.

5.2 Data Collection

To collect the required data, we implemented a simple Android app. Nine participants were asked to perform a process of picking up the phone then type. One of those participants

is considered the owner, and thus his data will be called the owner data sets, while the remaining collected data is called participants data sets. This process was repeated 8 different times for each participant except for the owner, who repeated it 60 times. The app collects and writes to separate files the acceleration on X, Y, and Z and the rotation angles yaw, pitch, and roll along with a time-stamp. In addition to these six elements, we also calculated and added the combined acceleration with gravity gA and the combined acceleration without gravity A. This is because in other studies, such as [15], the combined acceleration was used and it showed potential. Another reason for adding the combined acceleration is to check if it can replace the individual acceleration values. A and gA were calculated using the following formulas:

gA: $\sqrt{(X^2 + Y^2 + Z^2)}$

A: to calculate A these steps are taken [34]:

1. Calculate current gA and assign it to variable AccelLast
2. Calculate new gA (when a new value is requested)
3. Calculate delta as: gA - AccelLast
4. Calculate A as: A * 0.9 + delta (Initial value of A is 0)

The acceleration on X, Y, and Z, the combined acceleration gA and A, and the rotation angles will be called elements from now on. Each file would include the sensor data for one single pick-up or one single interaction. The following example shows what one measurement looks in a file. Typically, the file would have multiple measurements depending on the duration of the movement, and each line represents the values of the elements which were captured at time interval.

Example 5.2.1:

X	Y	Z	gA	A	Yaw	Pitch	Roll	Timestamp
-1.072	3.064	7.814	8.462	-1.701	0.301	-0.254	-0.025	1461346569331

Data collection was done at 10Hz (every 100ms) for the pick-up motion. This frequency proved to be fast enough to build a good model for training and testing without any noticeable noise. For example, trying to collect data every 5ms produced a very noisy signal with no added value. The reason behind this noisy signal is because the sensors constantly give different values. For example, if a phone is on a table, the acceleration on any axis will keep changing between 2 or 3 values very fast. As for micro-movement, collection was done at 100Hz (every 10ms). This is because the changes happen so frequently when typing and collecting data at 10Hz was not enough to capture data that truly represents the movement.

5.3 Normalization

A main issue with the collected orientation data is that the three orientation angles are affected by the earth magnetic pole. As can be seen in Figure 4.2, the orientation angles are calculated based on the earth magnetic pole. Since we can not know before hand the direction a user would be facing when using the phone, the effect of the magnetic pole must be removed. In addition, the values of X, Y, and Z of a movement produced by an accelerometer on different phones are shifted. For example, two phones (Samsung Note and Samsung S7) were put next to each other on a table in a still position and each one reported different values of Z (9.78 and 10.19 respectively). However, since we are using one phone model, no normalization is needed for the acceleration values. Normalization is still needed for the orientation angles. The effect of the north pole can be handled by computing the base

value of an angle (the value when the phone is stable) and then normalizing the data based on that. To do the normalization, the data collected at the beginning (150 ms) is averaged. This average is then subtracted from all subsequent values. By doing this normalization, the effect of the magnetic pole or different phones will be removed because all new collected data will be shifted toward zero.

5.4 Pick-Up Motion

We will discuss now the steps we took to create the pick-up motion authentication mechanism. First, we will talk about the training and creation of a classification model. Then, we will talk about the testing of the classification model.

5.4.1 Training

The pick-up motion can be considered as a time-series data, and as mentioned in chapter 4, DTW is a suitable algorithm for classification when dealing with data which is time-dependent. We use the FastDTW algorithm by Stan Salvador for Java[1]. The only parameter that can be adjusted is the distance function. A distance function is a function that defines the distance between elements of a set as a non-negative real number. it basically provides a way to measure how close two elements are [58]. FastDTW provides three alternatives: Manhattan, Euclidean, and Binary. Using the binary one will cause the final warp distance to either be 1 (equal) or 0 (otherwise). This does not suit our needs because the goal is finding the similarity between two data series and not whether they are equal or not. Euclidean and Manhattan distance resulted in very similar results, with differences around 0.05%. The most important difference between Euclidean and Manhattan is that if two points are close on most variables but discrepant on one of them, Euclidean distance will exaggerate that discrepancy, whereas Manhattan distance will ignore it. Thus, Euclidean distance was used as to make sure than the FAR is as low as possible.

The training for the pick-up motion consists of three phases: in phase one, part of owner's data will be processed via the FastDTW for the purpose of finding a threshold that can be used to judge whether new input belongs to the owner or to an intruder. In phase two, a different part of owner's data will be compared against phase one data via the FastDTW. The purpose of this comparison is to evaluate the accuracy of the threshold created in the previous phase. In phase three, part of the other participants' data will be compared against phase one data via the FastDTW. Again, the purpose of the comparison is to check the accuracy of the threshold created in phase one.

The reason behind using this method for training is because of the nature of DTW algorithm. DTW does not create a classification model the way other algorithms, such as SVM, do. Recall that DTW simply compares two time series and outputs a warp distance. The smaller the distance the more similar the two time series are. Another reason is that Conti et al. used similar approach in [4] and achieved good results. The training will conclude with a set of rules that summarize the results of the training. These rules will be used when new input from a user is processed to determine whether this user is the owner or not.

Phase 1: The first phase of training was done using 20 files (phase one training files) from the owner data sets. Recall that each file contains data for 8 elements. For the purpose of explaining, we will consider that we have 1 element only. The first file ($i = 1$) of those 20 files is compared with the remaining 19 files through the DTW algorithm. This produced 19

[1] https://github.com/medined/fastdtw.

different warp distances (WD). The average (Avr_i) and standard deviation (SD_i) of those 19 WD are calculated where i is the number of the file used. The procedure is repeated with the remaining files for a total of 20 Avr and 20 SD. The average (Avr_{total}) of those 20 averages is then calculated along with the standard deviation (SD_{total}). Then a base threshold was computed as: $BaseThreshold = Avr_{total} + SD_{total}$. In reality, we would have 8 different base thresholds, one for each element.

Phase 2: The Second phase of training used another 20 files (phase two training files) from the owner data sets. Each file was compared with the 20 phase one training files through the DTW algorithm. This procedure generated 20 different WD per file. Each of these 20 WD is compared with the $BaseThreshold$. The purpose of this comparison is to check if this $BaseThreshold$ can serve as the deciding point for accepting/rejecting a user. If it was indeed a good measure, then all the 20 previously calculated WD, or most of them, should be equal or smaller than the $BaseThreshold$, i.e., pass it. For this purpose, another 2 simple thresholds were created: the High Passing Threshold (HPT) represents a strict limit and is set to 15, and the Low Passing Threshold (LPT) represents a less strict limit and is set to 10. The idea behind those two thresholds is to check the number of times the newly calculated WD would pass the $BaseThreshold$. For example, if 18 of those WD have passed the $BaseThreshold$ then we consider the user as the owner with high confidence, or if 11 of those WD passed the $BaseThreshold$ then w consider the user as the owner with less confidence than before. However, that was not the case. A high number of those WD did not pass the $BaseThreshold$. This meant that the $BaseThreshold$ needs adjustments. Table 5.1 shows the result of the comparison.

Table 5.1: The results of comparing the 20 WD of each element computed in second phase against the $BaseThreshold$ - The higher the better

Element	HPT	LPT
X	37%	84%
Y	6%	37%
Z	6%	26%
gA	2%	6%
A	93%	97%
Yaw	0%	2%
Pitch	48%	97%
Roll	97%	97%

Phase 3: Third phase of training used 32 files from the participants (excluding the owner). Same as before, each file was compared to 20 phase one training files through the DTW algorithm. This comparison generated 32 different WD per file. Those 32 WD were compared with the $BaseThrehold$ and the results are shown in table 5.2. It is important to mention here that in this case it is needed that the WD do not pass the $BaseThreshold$ because we are dealing with non-owner data. The table indicates good results when dealing with non-owner. Thus, the $BaseThreshold$ needs adjustments so the owner can be recognized while keeping the good results of non-owner data.

Adjusting the $BaseThreshold$ is straightforward. The procedures in phase 2 and 3 are repeated while increasing the $BaseThreshold$ one at a time for 20 times. The following figures shows how incrementing the $BaseThreshold$ affects the acceptance percentage of the calculated WD.

Table 5.2: The results of comparing the 32 WD of each element computed in third phase against the *BaseThreshold* - The lower the better

Element	HPT	LPT
X	0%	0%
Y	0%	5%
Z	0%	0%
gA	0%	0%
A	0%	5%
Yaw	0%	5%
Pitch	25%	55%
Roll	65%	90%

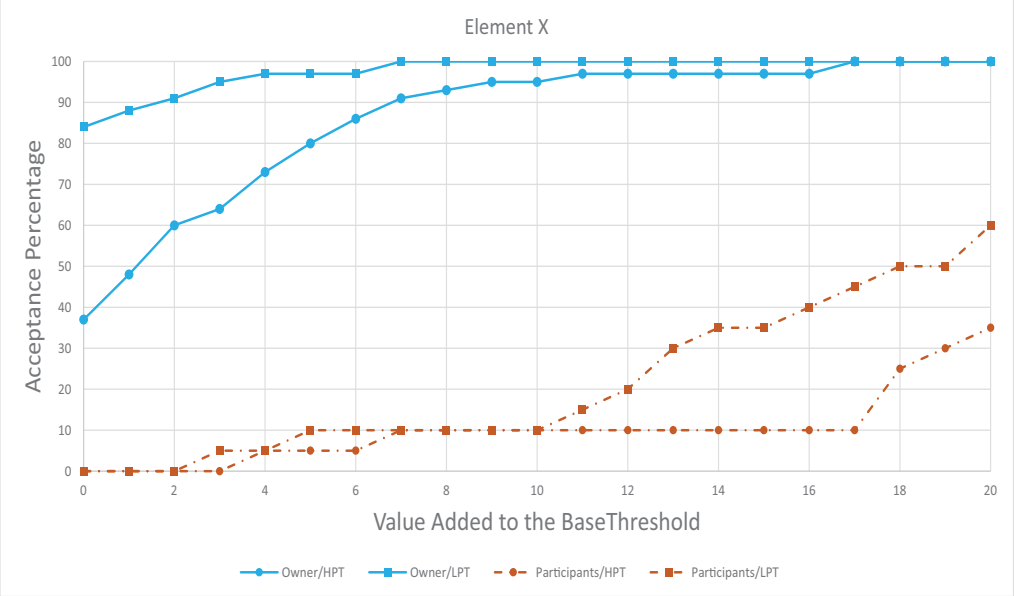


FIGURE 5.1: The effects of adjusting *BaseThreshold* on acceptance rate of the WD for element X

As we can see, incrementing the *BaseThreshold* improved the acceptance rate for the owner while keeping it low for non-owner. However, this only occurred with elements X, Y, Z, and A and with different level off effect. Table 5.3 shows the results of the process (some rows were deleted since the increasing of the *BaseThreshold* would yield no improvement after certain levels). The table shows that elements Yaw, Pitch, and Roll can not be used since the acceptance rate of the owner is very low even when increasing the *BaseThreshold*. On the other hand, elements X and A have the best results. Elements X and A can predict the owner with 91% and 93% accuracy using the LPT and HPT respectively while keeping the acceptance rate of non-owner at 0%. Elements Z and Y can predict the owner with 73% and 33% accuracy respectively using the HPT while keeping the acceptance rate of non-owner at 0%.

Following is a summary for the observations made of analyzing the results in table 5.3:

- Elements X, Y, Z, and A are the best suitable to be taken into account.

Table 5.3: The results of incrementing the *BaseThreshold* on the acceptance of the computed WD for all elements.

		HPT		LPT	
Element	Value Added to The BaseThreshold	Owner	Others	Owner	Others
X	0	37%	0%	84%	0%
	1	48%	0%	88%	0%
	2	60%	0%	91%	0%
	3	64%	0%	95%	5%
Y	0	6%	0%	37%	5%
	1	20%	0%	46%	10%
	2	24%	0%	60%	20%
	3	33%	0%	68%	20%
	4	40%	5%	71%	25%
Z	0	6%	0%	26%	0%
	1	13%	0%	35%	0%
	2	20%	0%	48%	0%
	3	26%	0%	53%	0%
	9	71%	0%	73%	10%
	10	73%	0%	77%	10%
	11	73%	5%	80%	20%
gA	0	2%	0%	6%	0%
	1	6%	0%	6%	0%
	2	6%	0%	6%	10%
	3	6%	5%	6%	10%
A	0	93%	0%	97%	5%
	1	97%	5%	97%	10%
	2	97%	10%	97%	20%
	3	97%	20%	97%	20%
Yaw	0	0%	0%	2%	5%
	1	0%	0%	15%	5%
	2	0%	0%	35%	5%
	3	0%	0%	53%	5%
	6	2%	5%	95%	10%
Pitch	0	48%	25%	97%	55%
	1	97%	45%	97%	55%
	2	97%	50%	97%	60%
	3	97%	55%	97%	65%
Roll	0	97%	65%	97%	90%
	1	97%	65%	97%	90%
	2	97%	75%	97%	90%
	3	97%	80%	97%	100%

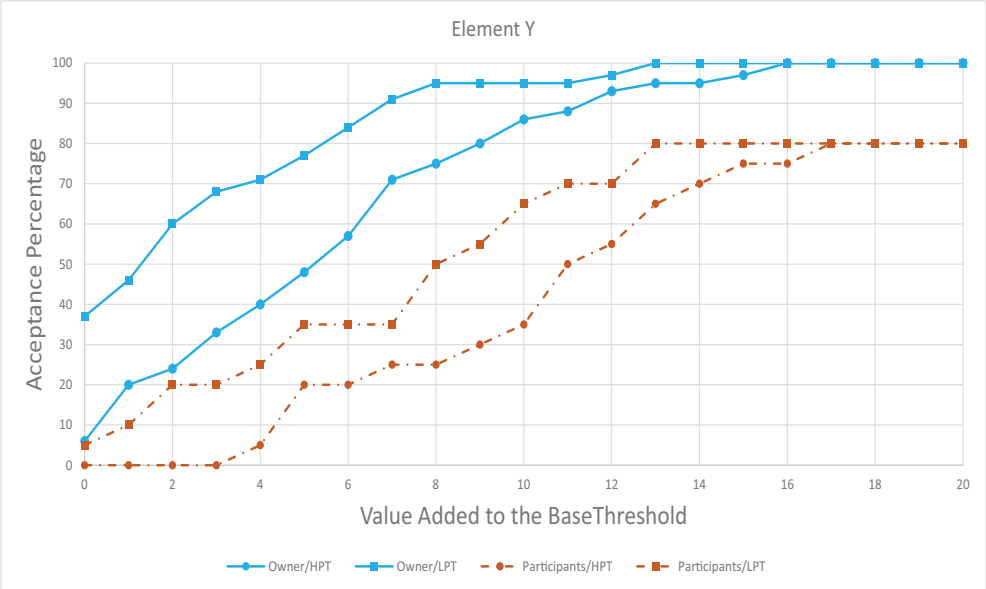


FIGURE 5.2: The effects of adjusting *BaseThreshold* on acceptance rate of the *WD* for element Y

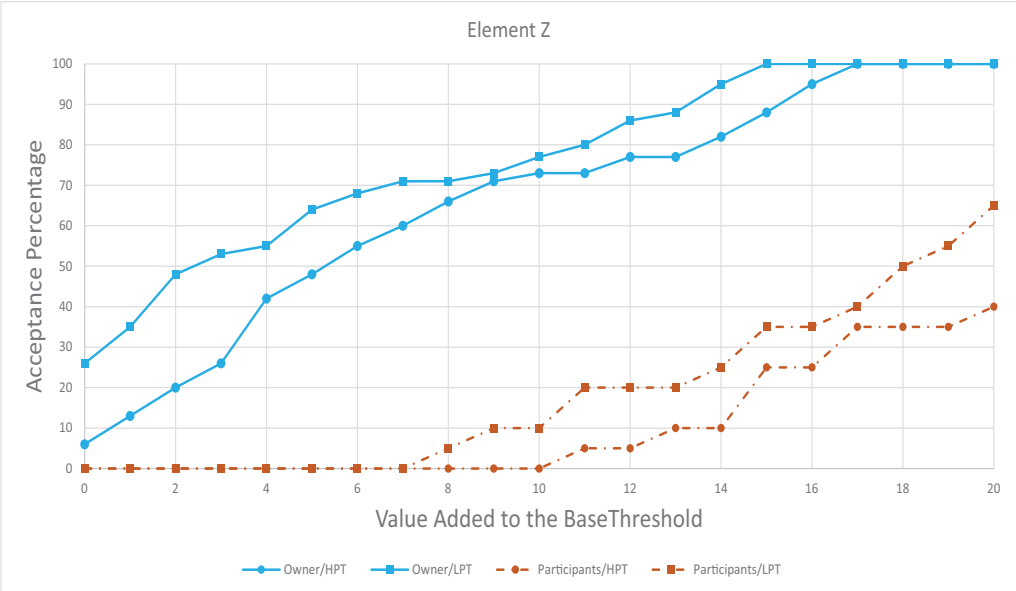


FIGURE 5.3: The effects of adjusting *BaseThreshold* on acceptance rate of the *WD* for element Z

- Some elements are better used with the low passing thresholds and other are better used with the high passing threshold.
- The value added to the base thresholds varies. This is probably because each element is affected differently by the pick-up motion. For example, the acceleration on Z is more similar between the owner and non-owner than the acceleration on Y, and thus

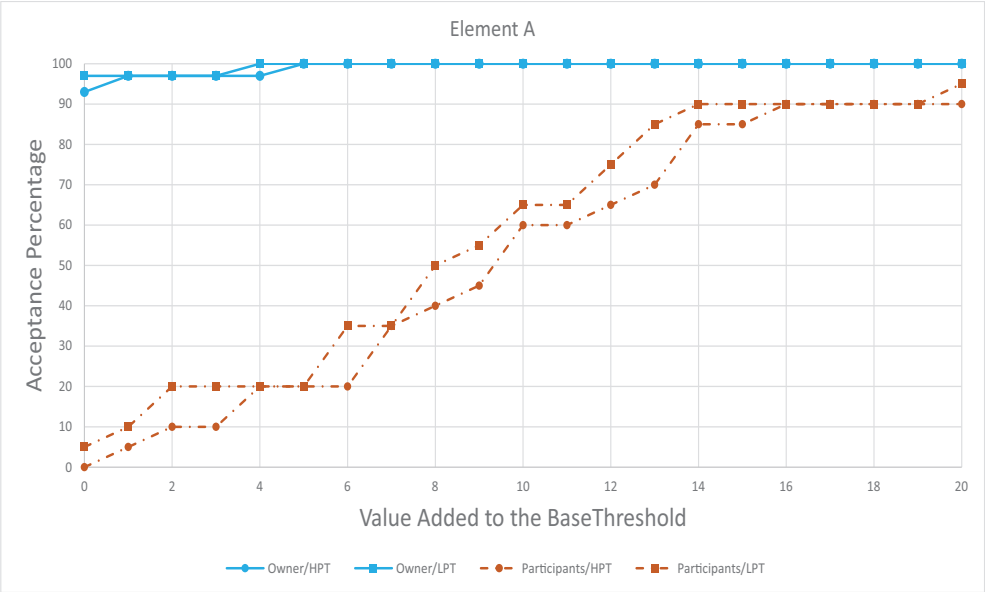


Figure 5.4: The effects of adjusting *BaseThreshold* on acceptance rate of the *WD* for element A

Table 5.4: The preferred "Passing Threshold" and "Added Number to the Base Threshold" along with corresponding acceptance rate acquired from the tests

Element	Preferred Passing Threshold	Preferred Added Number to the Base Threshold	Owner Acceptance Rate	Others Acceptance Rate
X	10	2	91%	0%
Y	15	3	33%	0%
Z	15	10	73%	0%
A	15	0	93%	0%

the extra number needed to differentiate between owner and non-owner wrap distances is different (10 in the case of Z and 3 in the case of Y).

Table 5.4 shows the chosen passing thresholds and the chosen numbers to be added to the base threshold. It is important to mention that the thresholds were chosen to have the lowest FAR possible.

Finally, the following rules were created for taking a decision about the authenticity of the person who picks up the phone. These rules are based on the data in table 5.3.

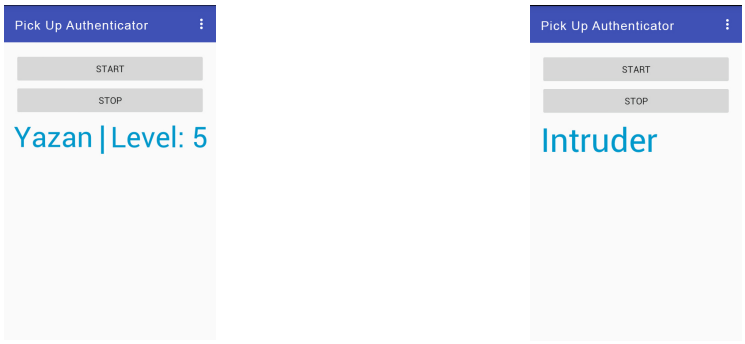
1. If elements (X, Y, Z, A) were accepted, then the user is considered the owner with very high probability and assigned an initial confidence level of 6.
2. If elements (X, Y, A) or (X, Z, A) were accepted, then the user is considered the owner with high probability and assigned an initial confidence level of 5.
3. If elements (X, A) were accepted, then the user is considered the owner with medium probability and assigned an initial confidence level of 4.
4. If elements (X, Y, Z) or (A, Y, Z) were accepted, then the user is considered the owner with medium probability and assigned an initial confidence level of 3.

5. If elements (X, Y) or (X, Z) or (A, Y) or (A, Z) were accepted, then the user is considered the owner with low probability and assigned an initial confidence level of 2.
6. If elements (Y, Z) were accepted, then the user is considered the owner with very low probability and assigned an initial confidence level of 1.
7. If none of the previous conditions are satisfied then the user is considered as intruder.

5.4.2 Testing

To test the effectiveness of the system, which was implemented as an Android app, two kind of tests were performed: the first one is with the owner and the second one is with other participants. The tester would pick up the phone a number of times and in each time the app would display whether the tester was identified as intruder or as the owner and with the corresponding level.

The testing procedure is as follows: the participant would press "Start" button, pick up the phone, then press "Stop" button. The "Start" button initiates data gathering, and "Stop" button stops it. The sensor data is saved to a file called "Current User". Next step is to calculate the WD between the "Current User" file and each of the 20 phase one training files. This calculation will result in 20 different wrap distances per element. Each of these wrap distances is compared with the corresponding threshold (base threshold + preferred added number). If the number of times a warp distance is less than threshold equals or more than the passing threshold then the element is considered accepted, other wise it is not. For example, if the number of warp distances calculated for element Z which are less than the threshold (in this case it is "Base Threshold + 10" as mentioned in table 5.4) is equal or more than its "Passing Threshold" (in the case of element Z it is 15) then the element is considered passed. After performing this test for the 4 elements (X, Y, Z, and A), a decision is taken based on the rules mentioned earlier. Figure 5.5 shows screenshots of the App after the owner and an intruder had picked up the phone.



(a) Owner (b) Intruder

FIGURE 5.5: Screenshots of the App after the owner (a) and an intruder (b) picked up the phone

To calculate the FRR, the owner performed the test 120 times. To calculate the FAR, 8 participants were asked to pick up the phone 10 times each. The results are in chapter 6.

5.5 Micro-Movements

Similar to the previous section, we will talk now about the steps we followed to create the micro-movements authentication mechanism. First, we will discuss the training which

consists of two round and each one includes features extraction and selection, and parameters adjustment. Then, we will talk about testing of the classification model.

5.5.1 Training

For micro-movement authentication, Support Vector Machine (SVM) was chosen as the training algorithm. SVM depends on expressing data by features. For example, accelerometer data collected while participant is performing a motion can be expressed by features such as average, amplitude, and energy. A Major advantage of SVM, in addition to the ones mentioned in chapter 4, is that SVM works better than other algorithms when: (1) training samples are few, and (2) 2-3 classes are being used. Creating a model for the micro movements is done using the LibSVM[2] library for support vector machine. This library was chosen due to its wide popularity.

First Round: In the first round of training, a window of approximately 2560ms was chosen to extract features from. The reason for choosing this length is be able to perform Fourier analysis on the data. Since data collection was done at 100Hz, this means that 2560ms is represented by 256 measurement, and Fourier analysis can only be performed on data with power of 2 length. The FFT (Fast Fourier Transform) was used on each window and it produced 256 components for each window. 10 windows were taken from owner training files and another 10 windows from other participants with 128 lines overlapping between consecutive windows. This 50% overlapping is because it showed better results in [41]. The reason behind choosing 10 windows is to train the SVM using only 15 seconds of user interaction. The following features were calculated for all the gathered and computed elements: the acceleration on X, Y, and Z, the combined acceleration gA and A, and the three rotational angles yaw, pitch, and roll:

- Energy: the sum of the squared discrete FFT component magnitudes of the signal.
- Magnitude Average: the average of the computed magnitudes of the FFT components. The magnitude is calculated using the following formula: (2/WindowFrame)*IMABS(FFT Component) where WindowFrame = 256 and IMBAS returns the absolute value of a complex number.
- Magnitude Total: the total sum of the computed magnitudes of the FFT components.
- Peak Amplitude Average: the average of the peak amplitude of the computed magnitudes of the FFT components. The peak amplitude is calculated using the following formula: $\sqrt{2} * IMBAS(FFTComponent)$.
- Peak Amplitude Total: the total sum of the computed peak amplitudes of the FFT components.
- The amplitude of the sum of all the FFT components.
- Root Mean Square (RMS): RMS is the square root of the arithmetic mean of the squares of a set of numbers.
- Standard Deviation.
- Mean.
- Average of the Absolute Deviations.
- Amplitude.

[2] https://www.csie.ntu.edu.tw/~cjlin/libsvm/

- Number of positive peaks: A positive peak is a point where the previous and next points are smaller.
- Number of negative peaks: A negative peak is a point where the previous and next points are bigger.
- Zero Cross Rate (ZCR): The number of times the signal switched from positive to negative or vise-versa.
- Histogram: this is used to represent the distribution of numerical data. To do this, a range of values between -2 and 2 with a step of 0.2 was used and then the number of values that fall into each interval is counted.
- Max Value.
- Min Value.
- Mode: The most frequently occurring, or repetitive, value.
- Percentile at 25% and 75%.

This totaled in 20 features per element and 160 features for all elements. Naturally, not all these features will provide a unique characteristics that can be used in the SVM. To distinguish the valuable features a python script[3] is used. This script calculate the importance score of each feature based on F-Score technique. F-Score is a simple technique which measures the discrimination of two sets of real numbers [43]. Given training vectors x_k, k = 1, . . . ,m, if the number of positive and negative instances are n_+ and n_-, respectively, then the F-Score of the ith feature is defined as:

$$F(i) \equiv \frac{(\bar{x}_i^{(+)} - \bar{x}_i)^2 + (\bar{x}_i^{(-)} - \bar{x}_i)^2}{\frac{1}{n_+ - 1}\sum_{k=1}^{n_+}(\bar{x}_{k,i}^{(+)} - \bar{x}_i^{(+)})^2 + \frac{1}{n_- - 1}\sum_{k=1}^{n_-}(\bar{x}_{k,i}^{(-)} - \bar{x}_i^{(-)})^2}$$

where $\bar{x}_i, \bar{x}_i^{(+)}, \bar{x}_i^{(-)}$ are the average of the ith feature of the whole, positive, and negative data sets, respectively; $\bar{x}_{k,i}^{(+)}$ is the ith feature of the kth positive instance, and $\bar{x}_{k,i}^{(-)}$ is the ith feature of the kth negative instance. The larger the F-Score is, the more likely this feature is more discriminative. According to the authors of [43], F-Score outperformed other techniques, such as random forests, in feature selection. According to this test, the following features were chosen for training: Histogram of X, RMS of X, ZCR of Yaw, Min of A, and Min of Pitch. Those 5 features were then sorted into one file for further processing before being fed as a training data for the SVM. Tables 5.5 and 5.6 show the 15 highest scores of applying F-Score to the training data and the estimated accuracy respectively.

The next step is to scale the data. The main purpose of scaling data before processing is to avoid attributes in greater numeric ranges. Other purpose is to avoid some types of numerical difficulties during calculation [23]. Scaling the data was done using LibSVM scaling tool. The range of scaling is [-1, +1] because some values are negative and it is recommended by the authors of the LibSVM to use this range in such a case, otherwise a scaling range of [0, +1] could be used.

To choose the right parameters' values for C and Gamma for each kernel, a grid search was used which incorporates a k-fold cross-validation. A grid search is simply an exhaustive searching through a manually specified subset of the parameters space. The reasons why

[3]https://www.csie.ntu.edu.tw/ cjlin/libsvmtools/fselect/fselect.py

Table 5.5: F-Score of the features calculated for the micro-movement classification model in round 1

Feature	Score
Histogram of X	13,802273
RMS of X	3,990075
ZCR of Yaw	2,331162
Min of A	2,204604
Min of Pitch	2,130131
Energy of X	1,859124
RMS of A	1,764267
Percentile 25 of A	1,595735
Average of A	1,562187
Energy of A	1,558623
ZCR of X	1,48494
ZCR of gA	1,472054
Mode of A	1,428601
Amplitude of the Sum of the FTT components of A	1,410658

Table 5.6: Estimated accuracy with respect to the number of features in round 1

Number of Features	Estimated Accuracy
160	47.05880%
80	47.05880%
40	47.05880%
20	47.05880%
10	47.05880%
5	100%
2	100%

a grid search was used over other types of methods are because it is easy to use, achieves best results, and doesn't hinder performance since training samples are few. The goal of a grid search is to identify good C and Gamma so that the classifier can accurately predict unknown data. In the case of a k-fold cross-validation grid search, the training set is divided into k subsets of equal size. Sequentially one subset is tested using the classifier trained on the remaining k-1 subsets using C and Gamma obtained from the grid. Thus, each instance of the whole training set is predicted once so the cross-validation accuracy is the percentage of data which are correctly classified [29]. The best C and Gamma are those who produced the highest cross-validation accuracy.

The grid search was executed on different kernel types, all of them produced an estimated accuracy rate of 100%. Hence, Choosing the right kernel required experimenting with the various types of kernels until the one that produces best rates is found. Linear kernel was tried first and while it produced good FAR, the FRR was very high and as a result this kernel was excluded. The remaining three kernels were tested and the RBF produced better results with a small margin. Subsequently, RBF is used to create the SVM model.

We performed tests on the created model and the results were not good (more on this in Chapter 6). Thus, we commenced a second round of feature selection and training.

Second Round: In this second round we repeated the same procedure as before, and so

we will only mention things that are different from the previous attempt. In this attempt, we only considered the acceleration on X, Y, and Z. The orientation angles were disregarded. The reason for this is because the orientation angles did not show good potential and the acceleration alone provided good results in the pick-up motion and in the pattern recognition (see section 5.6.3). In this attempt we started with a window of 5 seconds instead of 2.56 seconds. This is because we wanted to start the training and testing with relatively longer period and then, depending on the results, either increase or decrease the the duration. Also here we used 50% overlapping windows. The total windows used (i.e, training samples) are 6 for the owner (17.5 seconds) and 15 for non-owner (40 seconds). The following features were calculated for the acceleration on X, Y, and Z:

- Standard Deviation (SD).
- Amplitude.
- Number of Positive Peaks.
- Number of Negative peaks.
- Skew: skewness is a measure of the asymmetry of the probability distribution of a real-valued random variable about its mean.
- Average Absolute Deviation: the average of the absolute deviations of data points from their means (AAD).

The reasons for choosing these features are: (1) they are not affected by type of the phone (because these values do not change even if, for example, the acceleration on an axis was shifted), and (2) they produced good results when used in section 5.6.3. This resulted in 6 features per element and 18 total features. Again, the F-Score algorithm was used to determine the importance of these features. The result of applying the F-Score algorithm can be seen in tables 5.7 and 5.8.

Table 5.7: F-Score of the features calculated for the micro-movement classification model in round 2

Feature	Score
Negative peaks of Z	2.290891
AAD of Z	1.288442
Negative peaks of X	1.256739
Skew of Z	1.218504
SD of Z	1.022008
SD of X	0.715692
Positive peaks of Y	0.690170
AAD of X	0.684337
Amplitude of X	0.662507
Positive peaks of X	0.643998
Amplitude of Z	0.626706
AAD of Y	0.608538
Negative peaks of Y	0.553147
Positive peaks of Z	0.471781
SD of Y	0.398125
Skew of Y	0.253754
Skew of X	0.123615
Amplitude of Y	0.113825

Table 5.8: Estimated accuracy with respect to the number of features in round 2

Number of Features	Estimated Accuracy
18:	100.00000
9:	95.65220
4:	91.30430
2:	86.95650

Since the estimated accuracy was the highest when using all the features (see table 5.8), the classifier was built using all the 18 features (6 per element).

All different kernels expect of linear kernel produced an estimated accuracy of 100% when we used grid search with k-fold validation (linear kernel produced 92.3077% estimated accuracy). The three kernels were tested and the results are reported in chapter 6.

5.5.2 Testing

To test the micro-movement authentication, sensors' data from the owner and 7 participants was collected while typing. Random windows of 2560ms were taken from the data (to test the model of round 1). The five features that were selected in the first round of training were extracted from each window and used to compose one test file for a total of 220 test files (140 for owner and 80 for non-owner). The tests files then were scaled and fed to the prediction algorithm of the LibSVM library. The following examples shows what a test file looks like (for the round 1 model):

Example 5.3.1:

Before scaling:	1	1:0.154337908	2:16	3:241	4:-1.076258251	5:-0.61837955
After scaling:	1	1:-1.01344	2:-0.492063	3:0.951417	4:0.471176	5:-0.974612

As mentioned before, a histogram is a representation of the distribution of numerical data. It can be presented graphically or as a table such as table 5.9. However, the histogram appears as a single value in the previous example. This is because when the histograms were analyzed, we noticed that the data is concentrated in specific intervals. For example, when creating the histogram of X, we noticed that, in the case of the owner, the data is concentrated in the range -0.2 to 0.2. The feature value seen in the example is the sum of the number of values in this range. Table 5.9 shows how the histogram of the owner and three other participants looks like for acceleration on X.

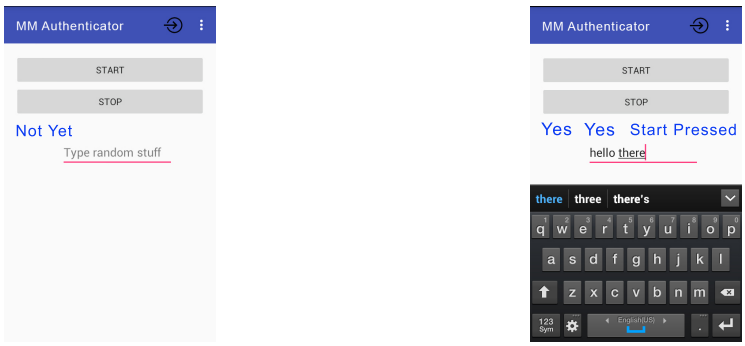
When the previous test file is fed to the prediction algorithm, it will either output *Yes* if it predicted that the data belongs to class 1 or *No* if it predicted that the data belongs to class 0.

Next, we repeated the test but this time for the second classification model we created. To do so, windows of 5 seconds length were extracted from the testing data and the features that were selected in the second round of training are calculated and fed to prediction algorithm.

Testing was done on a computer and also through a testing App. Figure 5.6 shows screenshots of this App. As can be seen in the previous image, there are two decisions displayed: the first one is for the 2.56 seconds (or 5 seconds) that started when the Start button was pressed and the second one is for the 2.56 seconds (or 5 seconds) that started 1.25 seconds (or 2.5 seconds) after the Start button was pressed. This is to make a decision on 50% overlapping data.

Table 5.9: The histogram for element X for the owner and three participants

	Owner	Participant 1	Participant 2	Participant 3
Bin	Frequency			
-1	0	0	0	0
-0.8	0	1	0	1
-0.6	0	6	0	2
-0.4	9	17	0	0
-0.2	20	11	0	2
0	90	18	0	0
0.2	137	21	0	0
0.4	0	26	15	2
0.6	0	10	102	14
0.8	0	26	113	34
1	0	49	26	140
1.2	0	21	0	23
1.4	0	20	0	15
1.6	0	20	0	15
1.8	0	4	0	3
2	0	2	0	2
More	0	4	0	3



(a) Startup (b) Decision Made

FIGURE 5.6: Screenshots of the App after startup (a) and after a decision was given (b)

5.6 Pattern Recognition

All the training and testing mentioned in previous sections were done in a controlled environment. This means that the starting and ending points of a motion (pick-up or micro-movements) were known. This was done to ensure that the data captured correctly represents the intended situation. However, in a real scenario, the starting and ending points of a motion will not be known beforehand. Therefore, there is a need for a pattern recognition systems that can correctly identify when a required motion occurs.

Detection of the pick-up motion can be tricky. The reasons for this are: (1) the pick-up motion is very short, ranging from 800ms to 1200ms, (2) it must be detected immediately otherwise the classifier will be fed with the wrong data, and (3) the pick-up motion is not periodic, which makes it harder to detect than other kinds on motions such as walking or climbing the stairs. There are two possibilities for specific-movement detection: first, a classifier can be trained with the pick-up motion in general, and second, thresholds can be identified by analyzing the data of the motion of various people.

5.6.1 Detection of Trigger Point

First, it is important for the pattern recognition system to know the potential starting point of the pick-up motion. This point can be distinguished by analyzing the data of various participants. By doing so, it was found that the pick-up motion starts with an increase (a peak) in the acceleration on Z around 80% of the time, and with a decrease in the acceleration on Z around 20% of the time. Thus, by monitoring the value of acceleration on Z, the pattern recognition can be triggered when this value pass certain threshold. Let us call the value of acceleration on Z when the phone is in a stable position (on a table) as "Base Z". When a pick-up motion occurs, the acceleration on Z increases by around 0.6 m/s^2 or decreases by around 0.3 m/s^2. Therefore, the first threshold point, which means a potential pick-up might have occurred, is: Z > (Base Z + 0.6) or Z < (base Z - 0.3).

In our experiment, Base Z was between 10.26 and 10.42 m/s^2, and thus the rule was: Z > 11 or Z < 10. It is important to mention that passing this threshold does not mean that a pick-up motion will follow. This is because passing the threshold could have been the result of the user, for example, moving his phone around on the table. As a result, more processing should take place after passing the threshold. Once this threshold is passed, sensor data is recorded for 1200 ms, which is the maximum time for a pick-up.

5.6.2 Creating Thresholds

The idea of using thresholds to detect a pick-up motion is to identify basic characteristics of the sensors data during the motion. For example, the highest or lowest value the acceleration on Z can reach. For this purpose, the pick data of all the participants was analyzed. Also, data of random movements (rolling the phone to the sides for example) was also analyzed. After the analysis, the following characteristics were found for the pick-up motion:

- Acceleration on X: the maximum value should pass 1.5 m/s^2 or the minimum value should go below -1.3 m/s^2.
- Acceleration on Y: the maximum value should pass 2 m/s^2 or the minimum value should go below -1.25 m/s^2.
- Acceleration on Z: the maximum value should pass 11 m/s^2 or the minimum value should go below 5 m/s^2.

When the Z Base rule mentioned previously is satisfied, sensor data is recorded for 1200 ms, and them simple calculations are done to determine the max and min values. A pick-up motion is detected if one of the conditions on X is met, one of the conditions on Y is met, and one of the conditions on Z is met.

5.6.3 Machine Learning

The second proposal to recognize the pick-up motion is by using machine learning. The idea is to do training for the pick-up motion in general. That is, using data from different users for training.

For this purpose, SVM was used again. Here, the two classes that the model is based on are simply "pick-up" and "non-pick-up". Again, the same data that was used in the previous attempt was used here. However, features needed to be extracted for the SVM to work properly. In this attempt, the features were calculated only on acceleration on X, Y, and Z. The reason for this (not calculating on the orientation angles or the combined acceleration) is because: (1) Acceleration on X, Y, and Z consistently showed more potential than the orientation angles, and (2) to make all the SVM calculation and file writing operations as simple as possible. This is because the number of times this test needs to be done can

Table 5.10: F-Score of the features calculated for the Pattern Recognition

Feature	FScore
Amplitude of Y	0.897686
SD of Y	0.883206
AVEDEV of Y	0.869310
Negative peaks of X	0.339713
Positive peaks of Z	0.314022
Amplitude of X	0.284564
Positive Peaks of X	0.208817
SD of X	0.187725
AVEDEV of X	0.162699
Negative Peaks of Y	0.127930
AVEDEV of Z	0.119816
Skew of Z	0.101950
Negative Peaks of Z	0.100806
Skew of Y	0.086368
Positive Peaks of Y	0.063984
SD of Z	0.062382
Amplitude of Z	0.052229
Skew of X	0.041696

Table 5.11: Estimated accuracy of the number of features for the Pattern Recognition

Number of Features	Estimated Accuracy
18:	100.00000
9:	100.00000
4:	92.50000
2:	87.50000

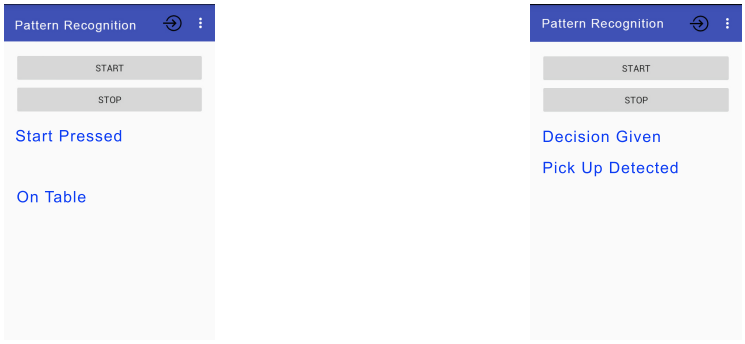
be high (every time a peak on Z is detected as mentioned earlier), and the result need to be calculated fast in order to send the data for further processing in case the motion was considered a pick-up. The features which were calculated are:

- Standard Deviation (SD).
- Amplitude.
- Number of positive peaks.
- Number of negative peaks.
- Skew: The degree of asymmetry of a distribution around its mean.
- Average Absolute Deviation: the average of the absolute deviations of data points from their means.

Those features were chosen because they are not affected by the change of the sensors' output between different phone models, and thus requiring less processing (no need for normalization).

After calculating these features, the FScore algorithm was used again to determine the importance of these features. Table 5.10 shows the score of each feature.

Table 5.11 shows the estimated accuracy with regard to the number of features used.



(a) Still Position (b) After a Pick-Up Motion

Figure 5.7: Screenshots of the App when the phone is in a still position on a table (a) and after a pick-up motion (b)

As seen in the table, 9 features can reach 100% accuracy for detecting a pick-up motion. Those features are the first 9 in table 5.10.

The next step is to choose a kernel. A grid search with K-Fold cross validation was done for each kernel to estimate the accuracy and choose the right values for the parameters C and Gamma. The results of this validation were as follows: Linear: 97.5%. Polynomial: 97.5%. Radial Based: 100%. Sigmoid: 95%. As a result, radial based kernel was chosen.

For testing, another Android app was developed. When the phone is on a table, the app would detect that and display a message "On Table". Once the Base Z rule mentioned earlier is met, the app would start recording accelerometer data for 1200 ms. Then, the 9 features will be calculated and then a prediction file would be created and sent to the SVM algorithm. The SVM would make a prediction and make a decision whether the motion belongs to the pick-up class or not. The decision then would be displayed on the screen. Figure 5.7 shows screenshots of the app when the phone is in a still position on a table and when a pick-up motion was performed and detected.

In addition to the app, pick-up files that were not used in the training were used for testing. Again, features were calculated and fed to the SVM classifier.

This chapter will include the results obtained from the assessments performed to test the pick-up motion authentication, micro-movements authentication, and the pattern recognition. The purpose of the testing was to determine the accuracy, mainly the FAR and FRR, of the authentication via pick-up and micro-movements and the accuracy of the pattern recognition.

6.1 Pick-Up Motion

This test aims to calculate the FAR and FRR of the pick-up motion based authentication. The testing starts with a user picking up the phone from a table. After the phone is picked up, the app will display the classification decision. The decision can either be (1) "Intruder" if none of the rules mentioned in section 5.4.1 were passed, or (2) "Owner: Level X" if at least one of the rules were passed.

The results of the FRR test are shown in table 6.1.

Table 6.1: Results of the test performed by the owner for the pick-up motion authentication

Confidence Level	Percentage %
6	42.5
5	20.8
4	0
3	16.6
2	16.6
1	3.3
Intruder	0

As can be seen from the table, the owner is never misidentified, that means that the FRR is 0%.

As for the FAR, none of the participants who performed the test were able to achieve a confidence level above intruder. After that, 5 participants were shown how the owner picks up the phone and then asked to imitate him. Again, none of them were able to achieve any confidence level above intruder. However, 1 participant (apart from the previous 8) was able to constantly achieve a confidence level of 1 by picking up the phone in a specific way. Also, he achieved confidence level of 5 one time. The level 1 constant result can be countered by merging it with intruder level, but this would mean that the FRR will increase from 0% to 3.3%.

We can argue that picking up the phone in a normal way by an intruder will always be detected. Thus, the FAR is 0%. However, an intruder with enough knowledge might be able to trick the system. For example, if an intruder knows that the phone authenticates users using the pick-up motion, then he or she can attempt to pick up the phone in different ways. Once he or she discovers a specific way to pick up the phone that can trick the mechanism, they can then repeat this motion and will be able to almost always trick the system. A solution for this is to perfectly identify realistic pick-up motions and ignore those that look suspicious (picking up the phone vertically for example). As a result, these weird movements will not be registered as a pick-up in the first place and will not be sent to the authentication mechanism.

With regard to the time needed to make the decision on the mobile phone, the process takes an average of 1.8 seconds to decide on the identity of the user.

6.2 Micro Movements

This test also aims to calculate the FAR and FRR of the micro-movements based authentication. Testing of the micro-movements authentication was done on a computer and via an app. The reason for doing the test on a computer is because there were already lots of captured data that was not used for training. Similar to training, features will be calculated on frames of 2.56 seconds for the model created in the first round and 5 seconds for the model created in the second round. An overlapping of 50% will also be done here. When performing live tests, and upon pressing the "Start" button in the app, sensors data will be recorded for a specific duration which depends on the model being tested as mentioned before to produce the first frame, and then another frame will be recorded that has a 50% overlapping with the previous one.The required features for each frame will then be calculated and then fed to the SVM classifier separately. For each frame, the classifier will either outputs *No* (user classified as non owner) or *Yes* (user classified as owner), and in the case of the live test, the result will be displayed on the screen.

After performing several tests for the first round model the results were not impressive. The FAR was around 29% and the FRR was around 32%. This was the reason for performing the second round of training mentioned in section 5.5.1. The testing in this round is done in the same manner but on windows of 5, 4, and 3 seconds. The results of this testing is shown in table 6.2.

Table 6.2: The FAR and FRR of the second classification model for the micro-movements

		Kernel		
Window Length (seconds)	Rate (%)	Radial-Based	Polynomial	Sigmoid
5	FRR	8.2	9.5	6.8
	FAR	12.3	9.2	12.3
4	FRR	16.3	16.7	14.2
	FAR	24.5	21.3	23.8
3	FRR	26.1	17.8	16.2
	FAR	31.2	26.4	28.3

We can see from the table that using a window of 5 seconds produces good results. Lowering this window to 4 or 3 seconds increases both the FAR and FRR noticeably. All of the kernels used produced good results. The Sigmoid kernel produced better FRR than the other two kernels while the Polynomial kernel produced better FAR than the other two kernels. When the mechanism is deployed in a real application the Sigmoid kernel can be used if the user is more concerned about usability. In contrast, the Polynomial kernel can be used if the user is more concerned about security.

Regarding processing time on the mobile phone, the classifier is able to give a decision in an average of 126.38 ms.

6.3 Pattern Recognition

This test aims to calculate the accuracy of detecting a pick-up motion (regardless of the identity of the user).

The purpose of the pattern recognition is crucial, it should be able to detect a pick-up motion exactly when it occurs so the data can then be sent for further processing. In the same time, it should not mistaken other types of movements, such as rolling the phone on the table, as a pick-up motion.

The aim of using the thresholds technique explained in section 5.6.2 was to achieve the goal of detecting a pick-up motion quickly. However, this method did not work quite well. When tested, there were many ways to fool it to think that a pick-up motion occurred. For example, rolling the phone to the left or to the right is detected as a pick-up. The problem is that if more thresholds were set as to not detect the roll as a pick-up, then actual pick-up motions might not be detected. Further tuning of the thresholds improved it but not enough to be a reliable method to detect a pick-up motion.

The SVM technique, which was explained in section 5.6.3, was tested via an app and on a computer using collected data. The pick-up data of participants, that was not used for training (32 files), were used for testing. The required features were calculated and then fed to SVM classifier. The classifier correctly identified all of the test data as pick-up motion.

The app works in a similar way. While the phone is on a table it would display "On Table". When it is moved and the Z Base rule mentioned in section 5.6.1 is met the app would display its decision (either "Pick Up" or "Some Movement"). A user picked up the phone 50 times, and the app was able to detect 48 of them. Then, the user moved the phone randomly (rolling, pitching, vertically) for 20 times each and the app never identified any of these movements as a pick-up. This means that the FAR is 0% and FRR is 2.5%. The time taken from the moment the Z Base rule is met until a decision is made is 997 ms on average.

Conclusions 7

This chapter will provide the answers for the research questions discussed in chapter 3, and an overview of future work that can extend and improve the work conducted in this thesis.

7.1 Discussion of Research Questions

Sub question 1: Can an owner of a mobile device be distinguished from other users by recognizing: (1) the way the owner picks up the phone, and (2) micro-movements of the phone during interactions?

As seen in figure 7.1, when two users, A and B, perform a pick-up motion twice, the captured acceleration on Z for both times for each user are similar to each other. In contrast, the captured acceleration on Z for one attempt for user A is different than its counterpart for user B. The same pattern appears with acceleration on X in figure 7.2. This means that the pick-up motion has the potential to be used to identify the owner of a device. That being said, classification models need to be created in order to perform this identification in real time.

Sub question 2: What kind of features can be extracted from collected sensors' data during the previously mentioned movements that have high discriminating characteristics that can be utilized to recognize the owner?

The goal of answering this question was to identify important features in the collected data of the two motions that can help build a good classification model. With regard to the pick-up motion, as was discussed previously, DTW was used as the classification algorithm. DTW did not require additional features to be calculated from collected sensors' data. Out of the 6 collected elements (acceleration on X, Y, and Z axes and orientation angles Yaw, Pitch, and roll) and 2 derived elements (combined acceleration with gravity gA and without gravity A), X and A appeared to be the most important factors in differentiating between the owner and other users. Z and Y also carry some distinction but not as good as X and A. The reason behind this might be because the acceleration on Z is more similar between different users because it always starts with a sudden peak and then a gradual decline. The acceleration on X is less similar between different users because it depends on the initial way the user carries the phone and how stable the phone stays during the pick-up motion. On the other hand, orientation angles did not provide good characteristics that can be used in classification. This is because the values of the orientation angles are not precise and not stable because the sensor is a derived one, in contrast to the accelerometer.

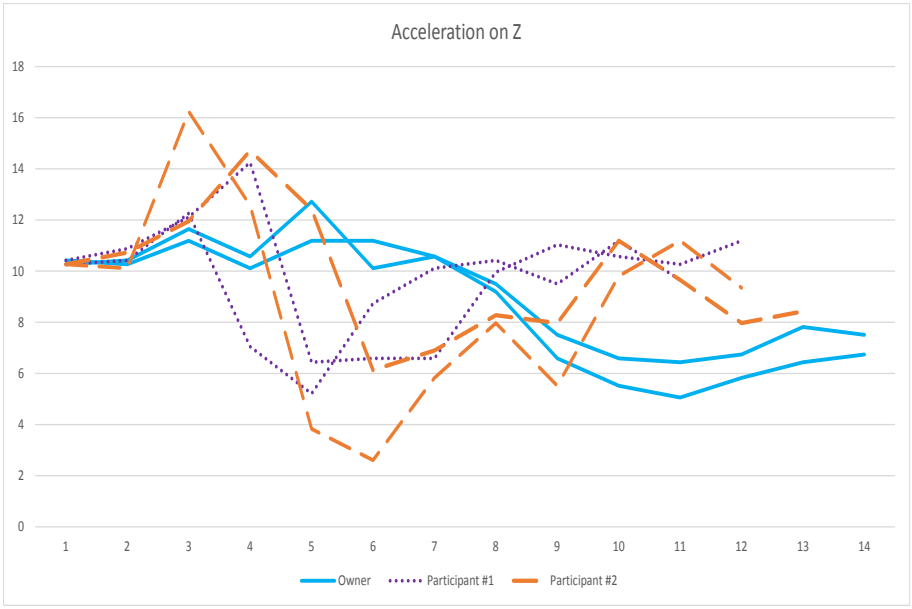


Figure 7.1: Acceleration on Z | Pick-Up Motion

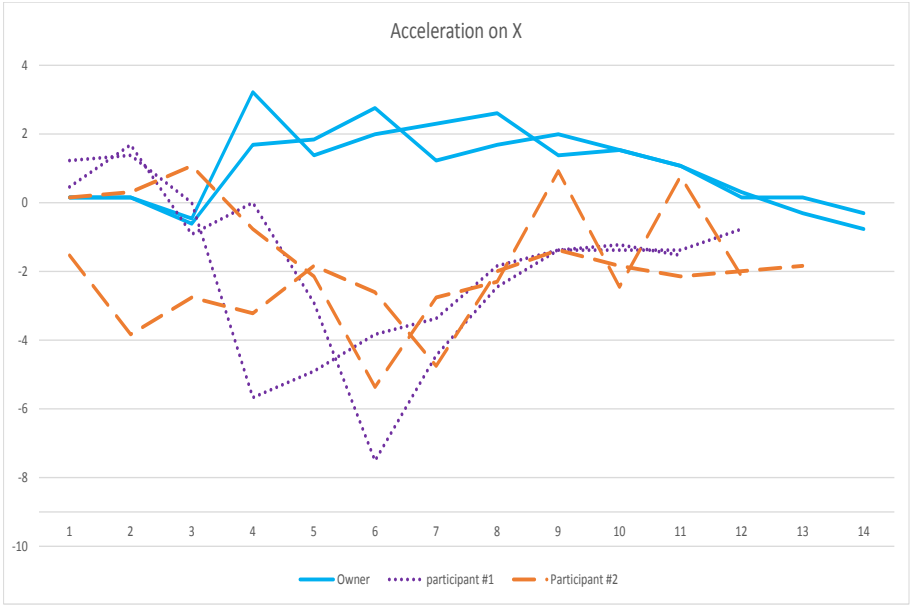


Figure 7.2: Acceleration on X | Pick-Up Motion

As for the micro-movements, features needed to be extracted from the collected data in order for the classifier to be able to build a good model that truly represents the owner. The following features were used to build the classification model (applicable to the acceleration on X, Y, and Z): Standard Deviation, Amplitude, Number of Negative Peaks, Number of Positive Peaks, Skew, and Average Absolute Deviation.

Sub question 3: What are the appropriate classifiers and parameters which work best

with these kind of movements?

Although only 1 classification algorithm was used per studied motion, both of them provided good results. The reason for choosing DTW for the pick-up motion is because this motion can be considered as a time-series and DTW is designed to work with time-dependent data. As for the micro-movements motion, SVM was chosen because in this experiment only 2 classes (owner and non-owner or pick-up and non-pick-up) were used, and SVM excels with binary classes. Also, SVM outperforms other algorithms when training samples are few. The intention behind using few training samples is to train the classifier using as low amount of data as possible. This will make training faster in real applications.

Apart from choosing the algorithms themselves, specific parameters need to be chosen for each one.

For the SVM, the first choice to make is whether to use regression or classification. Since the purpose of the model is to predict whether new data belongs to a predetermined class (owner/non-owner or pick-up/non-pick-up) rather than calculating new values, classification was the obvious choice. The second parameter to choose is the kernel. For the micro-movements, all the kernels produced an estimated accuracy of 100% when tested with k-fold cross validation (except for the linear one). However, after assessing each one with test data, Sigmoid kernel produced better FRR than the other two kernels while the Polynomial kernel produced better FAR than the other two kernels. For the pattern recognition, Radial Based was chosen because it produced the highest estimated accuracy during the k-fold cross validation test. The last two parameters to choose are C and Gamma. The best way to choose the right values for such parameters is via a grid search. The reason behind that is because there are huge amount of combinations for these two parameters. The grid search will simply scan through a defined range of values and do a cross validation for each combination. The one with the highest estimated accuracy is chosen. Table 7.1 shows the selected C and Gamma for each kernel.

Table 7.1: The selected C and Gamma for the Radial Based, Polynomial, and Sigmoid Kernels

Kernel	C	Gamma
Radial Based	128	0.0078125
Polynomial	0.5	0.5
Sigmoid	512	0.0078125

In the case of DTW, the only parameter that can be chosen is the distance metric. The used implementation provides three types: Manhattan, Euclidean, and Binary. using the binary implementation will cause the final warp distance to either be 1 (equal) or 0 (other wise). This does not suit our needs because the interest is finding the similarity between two data series and not whether they are equal or not. Euclidean and Manhattan distance resulted in very similar results, with differences around 0.05%. The most important difference between Euclidean and Manhattan is that if two points are close on most variables but discrepant on one of them, Euclidean distance will exaggerate that discrepancy, whereas Manhattan distance will ignore it. Thus, Euclidean distance was used as to make sure than the FAR is as low as possible. If an application focuses on usability more than security, then Manhattan distance can be used instead.

Although DTW had one parameter with two options to choose from, thresholds were created in order to derive a meaningful decision from the time warp measurements that the algorithm performs. The first one is the $BaseThreshold$. The purpose of this threshold is to have a reference for the owner so that WD computed on new data can be compared with this threshold. This threshold was calculated as:

$$BaseThreshold = Avr_{total} + SD_{total}.$$

Two more thresholds were created: High Passing Threshold HPT and Low Passing Threshold LPT. The purpose of these two thresholds is to check the number of times the newly calculated WD would pass the $BaseThreshold$. The HPT and LPT serve as strict and less strict thresholds respectively. HPT was set to 15 which represents 75% of the total owner's files used. LPT was set to 10 which represent 50% of the total owner's files used.

Sub question 4: How to correctly detect the pick-up motion at the right time so that the right data can then be sent to the authentication process?

As mentioned above, detecting a pick-up motion is crucial for the authentication mechanism to work properly. Two approaches were tried: using threshold and using machine learning. The first approach was a simple one which relied on creating a rules set of thresholds. When these rules are met, a pick-up motion was declared to be detected. This approach, however, did not yield reliable results. The second approach used machine learning to try to build a model that represents the pick-up motion. We used SVM for this purpose, where data from all participants were used to build a model for the pick-up motion in general. Indeed, the SVM approach yielded better results than the thresholds approach. The features that proved to be distinctive to the pick-up motion are: Amplitude of Y, SD of Y, AVEDEV of Y, Negative Peaks of X, Positive Peaks of Z, Amplitude of X, Positive Peaks of X, SD of X, and Average absolute deviation of X. As for the kernel, we used the Radial Based, and the corresponding parameters are C = 8 and Gamma = 0.0078125.

7.1.1 Summary

Sub question 1: Natural movements of phone usage such as pick-up motion and micro-movements can be utilized to authenticate owners of mobile devices.

Sub question 2: The acceleration on X, Y, and Z, and the combined acceleration A have good characteristics that can be exploited to create a model to authenticate the owner of mobile devices via pick-up motion. The acceleration on X and combined acceleration A provided better discriminant value than the acceleration on Y and Z.

The following features have good discriminating characteristics that can be used to build a classification model for the micro-movements authentication (applicable to the acceleration on X, Y, and Z): Standard Deviation, Amplitude, Number of Negative Peaks, Number of Positive Peaks, Skew, and Average Absolute Deviation.

Sub question 3: In the pick-up motion authentication, the Euclidean distance metric was chosen for the DTW algorithm because if two points are close on most variables but discrepant on one of them, Euclidean distance will exaggerate that discrepancy, and this led to a very good FAR. Another three thresholds were created to utilize the output of the DTW algorithm. These thresholds are: $BaseThreshold$, HPT, and LPT.

As for the micro-movements authentication, the three kernels (Radial Based, Polynomial, and Sigmoid) produced good results when tested. Sigmoid kernel produced better FRR than the other two kernels while the Polynomial kernel produced better FAR than the other two kernels. C and Gamma where chosen through a grid search.

Sub question 4: SVM-based pattern recognition was built to detect the motion of a pick-up movement. The features that had high discriminating characteristics for the pick-up motion (regardless of the owner) and used to build an SVM model are: Amplitude of Y, SD of Y, AVEDEV of Y, Negative Peaks of X, Positive Peaks of Z, Amplitude of X, Positive Peaks of X, SD of X, and AVEDEV of X. The Radial Based kernel is used and the corresponding parameters are C = 8 and Gamma = 0.0078125.

Main question: Can we provide continuous and implicit authentication from owner to mobile device with high accuracy while utilizing mobile devices' natural movements?

The main question of this thesis can be answered by saying that movements that are natural to a mobile phone usage can be used to authenticate the owner implicitly and continuously. The first movement, which is picking up the phone from a table, has the aspect that it usually precedes any other interaction with the phone, meaning that a user needs to pick up the phone from a table before starting to use it. As a result, this movement can serve as an initial security check. The other motion, which is the micro-movements of the phone during interaction, serves as a continuous security check, and can be extremely important when the user attempts to access a critical task, such as a banking app. The achieved accuracy of 0-3% FRR and 0% FAR for pick-up motion. The micro-movement authentication achieved 9.5% FRR and 9.2% FAR for the Polynomial kernel and 6.8% FRR and 12.3% FAR for the Sigmoid kernel.

7.2 Future Work

While the accuracies of the proposed authentication mechanisms are high, there are aspects of this work that can be improved to increase the confidence of the system.

7.2.1 Multi-Owner Experiment

The work on this thesis took into consideration only 1 of the participants as an owner and all the other participants as intruders. It would give better insight if the training and testing performed in sections 5.4 and 5.5 are repeated while switching between owner and intruders. For example, in the first round of training/testing, participant A would be considered the owner and the rest of the participants would be considered intruders. In this case, a model is created for participant A and tests will be made against this model. In the second round, participant B would be considered the owner while the rest of the participants, including A, would be considered intruders. Again, a different model is created for participant B and tests will be made against this model. This procedure can then be repeated several times. The goal of building different models for each participant is to shed more light on the: (1) rules built for the DTW for the pick-up motion, and (2) importance of features for the SVM for the micro-movements. For example, different participants might have different rules for the pick-up motion than the one discussed in 5.4.1 or different features importance than the one discovered in 5.5.1. This will result in an increased confidence in the authentication mechanisms.

7.2.2 1-Class Classification

In this work, multiclass classification was used for both the pick-up motion and micro-movements. As mentioned before, there were two classes throughout the training and testing: owner and non-owner. Doing this was possible because data was collected from multiple participants, and not only the owner. However, in a real situation, this might not be possible. Data from device owner can be easily collected to create the classification model, but it might not be convenient to ask other people to use the phone for data collection. A recommended solution is using 1-class classification. In 1-class classification, an algorithm will try to perform the training and create a classification model using only the data from one class. So in this case, the owner data, after being collected, will be used to create the classification model. When new unseen data is presented, the classifier will predict whether it belongs to the 1-class (owner) or not. SVM-based classification systems already exist for 1-class classification, such as the work of Schölkopf et al.[25] and Tax et al. [28].

7.2.3 Different Scenarios For More Flexibility

In this thesis, the pick-up motion was executed using a normal and simple way of picking up the phone. In this way, the participants would simply pick up the phone from a table in front of them. In order to increase the flexibility of the pick-up authentication, more positions for the pick-up can be studied. It would be more coherent if the training is done on more complex ways. An example situation would be if the participant is lying on a bed and the phone is on a table next to him/her. One solution is to build a system that can perfectly distinguish between two situations: the phone is on a table (P1) and the phone is in hand (P2). Then, when the system detects that the phone had shifted from P1 to P2, a pick-up motion is concluded to have been took place and the data can be logged.

The flexibility of the micro-movements-based authentication can also be improved by taking into consideration more practical situations. In this work, the micro-movements that instituted the test were extracted while the participants were typing. It would increase the scope of the system if micro-movements detected during other tasks, such as browsing, are also taken into consideration. Another way to improve the scope is by performing the data collection and testing while the user is walking. In this work, the micro-movements data were collected while the user was sitting. Two approaches can be used here to solve the problem: (1) users' data would be extracted once while they are sitting and again while they are walking and two models would be created accordingly, or (2) the changes that the walking movement would impose on the sensor data need to be isolated and then excluded from the overall data.

7.2.4 Resilience Against a Lego Robot Attack

As we saw in chapter 2, Serwadda et al. [27] designed a Lego robot that can execute attacks on touch-based authentication systems. This attack can increase the mean FAR of the majority of the classification algorithms by more than 100% of the mean FAR seen before the attack. It is important to study the possibility that a similar robot can attack against the movement-based authentication mechanism developed in this thesis. There are several points that needs to be taken into consideration: (1) can a Lego robot attack our movement-based authentication without knowledge of the algorithms used and the design of the App, (2) is there any population statistics that can be used to formulate the input for such an attack (population-driven attack), (3) how easy is it to formulate an attack if a user's data was stolen (user-tailored attack), (4) can a robot imitate a device owner (especially the pick-up motion) using a video of the owner while he/she performs the movement, and finally (5) what is the search space for a movement-based authentication system which in turn would determine the difficulty of performing brute force attack.

7.3 Final Conclusions

With the continuous increase of security breaches, lower trust of passwords amongst security professionals, and a higher demand for secure and user-friendly devices, contextual authentication has the potential to provide secure authentication mechanisms from users to personal devices without hindering users' experience. This can be addressed by utilizing various types of context such as location patterns and behavioral analysis.

This work provides two methods to authenticate users to their mobile devices. These methods do not require the user to perform any special tasks or input a PIN or a password. They simply take advantage of the movements of the phone when it is being used. The movements that were utilized are: (1) the pick-up motion of the phone from a table, and (2) micro-movements of phone during typing. These two mechanisms achieved reliable results that can lead to practical applications. However, there is still some room for improvement. Although the accuracy of both the authentication mechanisms proposed in this work is

high, this accuracy is only applicable to the user that was considered the owner during training and testing. This could mean that the accuracy might be lower for other people, or there might be other features than the ones selected that can be used to create a working classification model for other users. By repeating the training and testing while altering the user considered as the owner, the confidence in the two authentication mechanisms can greatly increase.

With regard to machine learning, multiclass classification was used in this thesis. Although multiclass classification usually produces better results than 1-class classification, it might not be practical in a real situation. Thus, it could improve the practicality of the authentication processes if 1-class classification is attempted to create the owner models.

Another limitation is with the way the micro-movements were collected. The sensor data was captured while the users were typing. However, it would increase the spectrum of when the authentication could take place if the experiment was also repeated with data captured while users were browsing web pages or images (i.e., flinging, pinching, and scrolling).

Finally, we can conclude that this thesis has presented further evidence that both the security and usability of mobile devices can be improved by utilizing the rich data mobile phone sensors provide to create a context-based authentication mechanism. In contrast to most of the related work, the work done here does not depend on the touch screen as the context factor, which was proven to be attackable by a Lego robot. Instead, it takes advantage of the movements of the phone to identify the owner. It is important to note that the used movements are natural to phone usage which means that the users are not required to perform any additional tasks to authenticate themselves. The two used movements are different in the sense that one of them (pick-up) usually precedes any other action on the phone, and the other one (micro-movements) happens during actual interaction with the phone which can serve as a way for continuous authentication.

This work takes us one step closer to the transition to contextual authentication and excluding passwords from our security practices and habits. With more security breaches everyday that damage reputation, ruin lives, and lead to financial losses, contextual authentication can strengthen the information security to give more protection to digital assets.

Bibliography

Papers

[1] A. Anjum and M. U. Ilyas. "Activity recognition using smartphone sensors". In: *2013 IEEE 10th Consumer Communications and Networking Conference (CCNC)*. IEEE, Jan. 2013, pp. 914–919. DOI: 10.1109/CCNC.2013.6488584.

[2] Cheng Bo et al. "SilentSense: Silent User Identification via Touch and Movement Behavioral Biometrics". In: *Proceedings of the 19th Annual International Conference on Mobile Computing & Networking*. MobiCom '13. Miami, Florida, USA: ACM, 2013, pp. 187–190. ISBN: 978-1-4503-1999-7. DOI: 10.1145/2500423.2504572. URL: http://doi.acm.org/10.1145/2500423.2504572.

[3] Keng-Hao Chang, Jeffrey Hightower, and Branislav Kveton. "Inferring Identity Using Accelerometers in Television Remote Controls". In: *Proceedings of the 7th International Conference on Pervasive Computing*. Pervasive '09. Nara, Japan: Springer-Verlag, 2009, pp. 151–167. ISBN: 978-3-642-01515-1. DOI: 10.1007/978-3-642-01516-8_11. URL: http://dx.doi.org/10.1007/978-3-642-01516-8_11.

[4] Mauro Conti, Irina Zachia-Zlatea, and Bruno Crispo. "Mind How You Answer Me!: Transparently Authenticating the User of a Smartphone when Answering or Placing a Call". In: *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*. ASIACCS '11. Hong Kong, China: ACM, 2011, pp. 249–259. ISBN: 978-1-4503-0564-8. DOI: 10.1145/1966913.1966945. URL: http://doi.acm.org/10.1145/1966913.1966945.

[5] Tao Feng et al. "Continuous Mobile Authentication Using Virtual Key Typing Biometrics". In: *Proceedings of the 2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*. TRUSTCOM '13. Washington, DC, USA: IEEE Computer Society, 2013, pp. 1547–1552. ISBN: 978-0-7695-5022-0. DOI: 10.1109/TrustCom.2013.272. URL: http://dx.doi.org/10.1109/TrustCom.2013.272.

[6] B. Huang and W. Kinsner. "ECG frame classification using dynamic time warping". In: *2*. IEEE, 2002, 1105–1110 vol.2. DOI: 10.1109/CCECE.2002.1013101.

[7] Hassan Khan, Aaron Atwater, and Urs Hengartner. "Itus: An Implicit Authentication Framework for Android". In: *Proceedings of the 20th Annual International Conference on Mobile Computing and Networking*. MobiCom '14. Maui, Hawaii, USA: ACM,

2014, pp. 507–518. ISBN: 978-1-4503-2783-1. DOI: 10.1145/2639108.2639141. URL: http://doi.acm.org/10.1145/2639108.2639141.

[8] Lingjun Li, Xinxin Zhao, and Guoliang Xue. "Unobservable Re-authentication for Smartphones." In: *20th Annual Network and Distributed System Security Symposium*. The Internet Society, 2013. URL: http://www.internetsociety.org/sites/default/files/02_1_0.pdf.

[9] Muhammad Shahzad, Alex X. Liu, and Arjmand Samuel. "Secure Unlocking of Mobile Touch Screen Devices by Simple Gestures: You Can See It but You Can Not Do It". In: *Proceedings of the 19th Annual International Conference on Mobile Computing & Networking*. MobiCom '13. ACM, 2013, pp. 39–50. ISBN: 978-1-4503-1999-7. DOI: 10.1145/2500423.2500434. URL: http://doi.acm.org/10.1145/2500423.2500434.

[10] Elaine Shi et al. "Implicit Authentication Through Learning User Behavior". In: *Proceedings of the 13th International Conference on Information Security*. ISC'10. Boca Raton, FL, USA: Springer-Verlag, 2011, pp. 99–113. ISBN: 978-3-642-18177-1. DOI: 10.1007/978-3-642-18178-8_9. URL: http://dl.acm.org/citation.cfm?id=1949317.1949329.

[11] H. Witte, C. Rathgeb, and C. Busch. "Context-Aware Mobile Biometric Authentication based on Support Vector Machines". In: *Emerging Security Technologies (EST), 2013 Fourth International Conference on*. IEEE, Sept. 2013, pp. 29–32. ISBN: 978-0-7695-5077-0. DOI: 10.1109/EST.2013.38.

[12] Yi Xu et al. "Virtual U: Defeating Face Liveness Detection by Building Virtual Models from Your Public Photos". In: *25th USENIX Security Symposium (USENIX Security 16)*. USENIX Association, Aug. 2016, pp. 497–512. ISBN: 978-1-931971-32-4. URL: https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/xu.

Articles

[13] John Aach and George M. Church. "Aligning gene expression time series with time warping algorithms". In: *Bioinformatics* 17.6 (2001), pp. 495–508. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/17.6.495.

[14] Heather Crawford, Karen Renaud, and Tim Storer. "A Framework for Continuous, Transparent Mobile Device Authentication". In: *Comput. Secur.* 39 (Nov. 2013), pp. 127–136. ISSN: 0167-4048. DOI: 10.1016/j.cose.2013.05.005. URL: http://dx.doi.org/10.1016/j.cose.2013.05.005.

[15] Mohammad Derawi and Patrick Bours. "Gait and Activity Recognition Using Commercial Phones". In: *Comput. Secur.* 39 (Nov. 2013), pp. 137–144. ISSN: 0167-4048. DOI: 10.1016/j.cose.2013.07.004. URL: http://dx.doi.org/10.1016/j.cose.2013.07.004.

[16] Marcos Faundez-Zanuy. "On-line signature recognition based on VQ-DTW". In: *Pattern Recognition* 40.3 (2007), pp. 981–992. ISSN: 0031-3203. DOI: http://dx.doi.org/10.1016/j.patcog.2006.06.007. URL: http://www.sciencedirect.com/science/article/pii/S0031320306002780.

[17] Steven Furnell, Nathan Clarke, and Sevasti Karatzouni. "Beyond the PIN: Enhancing user authentication for mobile devices". In: *Computer Fraud & Security* 2008.8 (2008), pp. 12–17. ISSN: 1361-3723. DOI: http://dx.doi.org/10.1016/S1361-3723(08)70127-1. URL: http://www.sciencedirect.com/science/article/pii/S1361372308701271.

[18] Toni Giorgino. "Computing and Visualizing Dynamic Time Warping Alignments in R: The dtw Package". In: *Journal of Statistical Software* 31.7 (2009), pp. 1–24. ISSN: 1548-7660. DOI: 10.18637/jss.v031.i07. URL: http://www.jstatsoft.org/v31/i07/.

[19] K. Gollmer and C. Posten. "Supervision of bioprocesses using a dynamic time warping algorithm". In: *Control Engineering Practice* 4.9 (1996), pp. 1287–1295. ISSN: 0967-0661. DOI: http://dx.doi.org/10.1016/0967-0661(96)00136-0. URL: http://www.sciencedirect.com/science/article/pii/0967066196001360.

[20] Hilmi Günes Kayacik et al. "Data Driven Authentication: On the Effectiveness of User Behaviour Modelling with Mobile Device Sensors". In: *CoRR* abs/1410.7743 (2014). URL: https://arxiv.org/pdf/1410.7743.

[21] Jennifer R. Kwapisz, Gary M. Weiss, and Samuel A. Moore. "Activity Recognition Using Cell Phone Accelerometers". In: *SIGKDD Explor. Newsl.* 12.2 (Mar. 2011), pp. 74–82. ISSN: 1931-0145. DOI: 10.1145/1964897.1964918. URL: http://doi.acm.org/10.1145/1964897.1964918.

[22] Elizabeth Mauch. "Using Technological Innovation to Improve the Problem-Solving Skills of Middle School Students: Educators' Experiences with the LEGO Mindstorms Robotic Invention System". In: *The Clearing House: A Journal of Educational Strategies, Issues and Ideas* 74.4 (2001), pp. 211–213. DOI: 10.1080/00098650109599193. URL: http://dx.doi.org/10.1080/00098650109599193.

[23] Lekha Bhambhu Minaxi Arora. "Role of Scaling in Data Classification Using SVM". In: *International Journal of Advanced Research in Computer Science and Software Engineering* 4 (2014), pp. 271–273. ISSN: 2277 128X. URL: http://www.ijarcsse.com/docs/papers/Volume_4/10_October2014/V4I10-0254.pdf.

[24] Stan Salvador and Philip Chan. "Toward Accurate Dynamic Time Warping in Linear Time and Space". In: *Intell. Data Anal.* 11.5 (Oct. 2007), pp. 561–580. ISSN: 1088-467X. URL: http://dl.acm.org/citation.cfm?id=1367985.1367993.

[25] Bernhard Schölkopf et al. "Estimating the Support of a High-Dimensional Distribution". In: *Neural Comput.* 13.7 (July 2001), pp. 1443–1471. ISSN: 0899-7667. DOI: 10.1162/089976601750264965. URL: http://dx.doi.org/10.1162/089976601750264965.

[26] Hojin Seo, Eunjin Kim, and Huy Kang Kim. "A novel biometric identification based on a user's input pattern analysis for intelligent mobile devices". In: *International Journal of Advanced Robotic Systems* 9 (2012). ISSN: 1729-8806. DOI: 10.5772/51319.

[27] Abdul Serwadda et al. "Toward Robotic Robbery on the Touch Screen". In: *ACM Transactions on Information and System Security* 18.4 (May 2016), 14:1–14:25. ISSN: 1094-9224. DOI: 10.1145/2898353. URL: http://doi.acm.org/10.1145/2898353.

[28] David M. J. Tax and Robert P. W. Duin. "Support Vector Data Description". In: *Mach. Learn.* 54.1 (Jan. 2004), pp. 45–66. ISSN: 0885-6125. DOI: 10.1023/B:MACH.0000008084.60811.49. URL: http://dx.doi.org/10.1023/B:MACH.0000008084.60811.49.

Technical Reports

[29] Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin. *A Practical Guide to Support Vector Classification*. Department of Computer Science, National Taiwan University, 2003. URL: http://www.csie.ntu.edu.tw/~cjlin/papers.html.

[30] Lawless Research. *Beyond the Password: The Future of Account Security*. TeleSign, 2016. URL: https://www.telesign.com/resources/research-and-reports/beyond-password-future-account-security/.

[31] Tao Wei et al. *Fingerprints On Mobile Devices: Abusing And Leaking*. FireEye Labs, 2015. URL: https://www.blackhat.com/docs/us-15/materials/us-15-Zhang-Fingerprints-On-Mobile-Devices-Abusing-And-Leaking-wp.pdf.

Books

[32] Ethem Alpaydin. *Introduction to Machine Learning*. Adaptive Computation and Machine Learning. The MIT Press, 2010. ISBN: 9780262012430.

[33] Paul S. P. Cowpertwait and Andrew V. Metcalfe. *Introductory Time Series with R*. Springer Publishing Company, Incorporated, 2009. ISBN: 0387886974, 9780387886978.

[34] Jeff McWherter and Scott Gowell. *Professional Mobile Application Development*. Wrox Press Ltd., 2012. ISBN: 1118203909, 9781118203903.

[35] Meinard Müller. *Information Retrieval for Music and Motion*. Springer-Verlag New York, Inc., 2007. ISBN: 3540740473.

[36] William H. Press et al. *Numerical Recipes in FORTRAN; The Art of Scientific Computing*. Cambridge University Press, 1993. ISBN: 0521437164.

[37] Phil Simon. *Too big to ignore: the business case for big data*. Wiley & SAS Business Series. Wiley, 2013. ISBN: 978-1-118-63817-0.

[38] J.J. Stapleton. *Security without Obscurity: A Guide to Confidentiality, Authentication, and Integrity*. EBL-Schweitzer. CRC Press, 2014. ISBN: 9781466592155.

In Books

[39] Stephanie A. C. Schuckers. "Liveness Detection". In: *Encyclopedia of Biometrics*. Ed. by Stan Z. Li and Anil Jain. Springer US, 2009, pp. 924–924. ISBN: 978-0-387-73003-5. DOI: 10.1007/978-0-387-73003-5_76. URL: http://dx.doi.org/10.1007/978-0-387-73003-5_76.

[40] Julio Angulo and Erik Wästlund. "Exploring Touch-Screen Biometrics for User Identification on Smart Phones". In: *Privacy and Identity Management for Life: 7th IFIP WG 9.2, 9.6/11.7, 11.4, 11.6/PrimeLife International Summer School, Trento, Italy, September 5-9, 2011, Revised Selected Papers*. Ed. by Jan Camenisch et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 130–143. ISBN: 978-3-642-31668-5. DOI: 10.1007/978-3-642-31668-5_10. URL: http://dx.doi.org/10.1007/978-3-642-31668-5_10.

[41] Ling Bao and Stephen S. Intille. "Activity Recognition from User-Annotated Acceleration Data". In: *Pervasive Computing: Second International Conference, PERVASIVE 2004, Linz/Vienna, Austria, April 21-23, 2004. Proceedings*. Ed. by Alois Ferscha and Friedemann Mattern. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 1–17. ISBN: 978-3-540-24646-6. DOI: 10.1007/978-3-540-24646-6_1. URL: http://dx.doi.org/10.1007/978-3-540-24646-6_1.

[42] Josep Maria Carmona and Joan Climent. "A Performance Evaluation of HMM and DTW for Gesture Recognition". In: *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications: 17th Iberoamerican Congress, CIARP 2012, Buenos Aires, Argentina, September 3-6, 2012. Proceedings*. Ed. by Luis Alvarez et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 236–243. ISBN: 978-3-642-33275-3. DOI: 10.1007/978-3-642-33275-3_29. URL: http://dx.doi.org/10.1007/978-3-642-33275-3_29.

[43] Yi-Wei Chen and Chih-Jen Lin. "Combining SVMs with Various Feature Selection Strategies". In: *Feature Extraction: Foundations and Applications*. Ed. by Isabelle Guyon et al. Springer Berlin Heidelberg, 2006, pp. 315–324. ISBN: 978-3-540-35488-8. DOI: 10.1007/978-3-540-35488-8_13. URL: http://dx.doi.org/10.1007/978-3-540-35488-8_13.

[44] Bendik B. Mjaaland, Patrick Bours, and Danilo Gligoroski. "Walk the Walk: Attacking Gait Biometrics by Imitation". In: *Information Security: 13th International Conference, ISC 2010, Boca Raton, FL, USA, October 25-28, 2010, Revised Selected Papers*. Ed. by Mike Burmester et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 361–380. ISBN: 978-3-642-18178-8. DOI: 10.1007/978-3-642-18178-8_31. URL: http://dx.doi.org/10.1007/978-3-642-18178-8_31.

[45] Nashad Ahmed Safa, Reihaneh Safavi-Naini, and Siamak F. Shahandashti. "Privacy-Preserving Implicit Authentication". In: *ICT Systems Security and Privacy Protection: 29th IFIP TC 11 International Conference, SEC 2014, Marrakech, Morocco, June 2-4, 2014. Proceedings*. Ed. by Nora Cuppens-Boulahia et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 471–484. ISBN: 978-3-642-55415-5. DOI: 10.1007/978-3-642-55415-5_40. URL: http://dx.doi.org/10.1007/978-3-642-55415-5_40.

[46] M. A. Sasse and I. Flechais. "Usable Security: What Is It? How Do We Get It?" In: *Security and Usability: Designing Secure Systems that People can Use*. Ed. by Lorrie Faith Cranor and Simson Garfinkel. O'Reilly Books, 2005, pp. 13–30. ISBN: 978-0-596-00827-7.

Online

[47] Bayometric. *False Acceptance Rate (FAR) & False Recognition Rate (FRR)*. [Online; accessed September 15, 2016]. 2016. URL: http://www.bayometric.com/blog/false-acceptance-rate-far-false-recognition-rate-frr/.

[48] Dan Moren. *Face Recognition Security, Even With A 'Blink Test,' Is Easy To Trick*. [Online; accessed April 1, 2016]. 2015. URL: http://www.popsci.com/its-not-hard-trick-facial-recognition-security.

[49] Elliot Williams. *Your Unhashable Fingerprints Secure Nothing*. [Online; accessed September 9, 2016]. 2015. URL: http://hackaday.com/2015/11/10/your-unhashable-fingerprints-secure-nothing/.

[50] Dimension Engineering. *A beginner's guide to accelerometers*. [Online; accessed May 31, 2016]. 2016. URL: http://www.dimensionengineering.com/info/accelerometers.

[51] Google. *Motion Sensors*. [Online; accessed May 31, 2016]. 2016. URL: https://developer.android.com/guide/topics/sensors/sensors_motion.html.

[52] Google. *SensorEvent*. [Online; accessed May 31, 2016]. 2016. URL: https://developer.android.com/reference/android/hardware/SensorEvent.html.

[53] Google. *Sensors Overview*. [Online; accessed April 1, 2016]. 2016. URL: http://developer.android.com/guide/topics/sensors/sensors_overview.html.

[54] Howard Baldwin. *Passwords are the weak link in IT security*. [Online; accessed September 9, 2016]. 2012. URL: http://www.computerworld.com/article/2493184/security0/passwords-are-the-weak-link-in-it-security.html?page=2.

[55] IDC. *Smartphone OS Market Share, 2015 Q2*. [Online; accessed June 6, 2016]. 2015. URL: http://www.idc.com/prodserv/smartphone-os-market-share.jsp.

[56] Keith Graham. *Moving Beyond 2-Factor Authentication With 'Context'*. [Online; accessed April 1, 2016]. 2014. URL: http://www.darkreading.com/endpoint/authentication/moving-beyond-2-factor-authentication-with-context/a/d-id/1317911.

[57] Information Technology Laboratory. *Introduction to Time Series Analysis*. [Online; accessed May 6, 2016]. 2015. URL: http://www.itl.nist.gov/div898/handbook/pmc/section4/pmc41.htm.

[58] Math.NET. *Distance Metrics*. [Online; accessed September 8, 2016]. 2016. URL: http://numerics.mathdotnet.com/Distance.html.

[59] Natasha Lomas. *Battery Attributes Can Be Used To Track Web Users*. [Online; accessed April 2, 2016]. 2015. URL: http://techcrunch.com/2015/08/04/battery-attributes-can-be-used-to-track-web-users/.

[60] Jakob Nielsen. *Usability 101: Introduction to Usability*. [Online; accessed July 26, 2016]. 2012. URL: https://www.nngroup.com/articles/usability-101-introduction-to-usability/.

[61] TeamsID. *Worst Passwords of 2015*. [Online; accessed April 1, 2016]. 2015. URL: https://www.teamsid.com/worst-passwords-2015/.

[62] Wikipedia. *Gait*. [Online; accessed April 2, 2016]. 2015. URL: https://en.wikipedia.org/wiki/Gait.

[63] Wikipedia. *Regression Analysis*. [Online; accessed April 15, 2016]. 2016. URL: https://en.wikipedia.org/wiki/Regression_analysis.