Continuous Authentication on a Smartwatch

M. Migliardi[1,2(✉)], M. Guerar[1,2], S. Marzio[1], and C. Ferrari[1,2]

[1] DEI - University of Padua, Padua, Italy
{m.migliardi,m.guerar,s.marzio,c.ferrari}@unipd.it
[2] CIPI - University of Padua, Padua, Italy
http://www.dei.unipd.it

AQ1

Abstract. The purpose of this work is to leverage two types of sensors, motion and optical, to create a continuous authentication system for smart devices such as smartwatches. The proposed solution is based on an Android application that uses the accelerometer and gyroscope to measure movements and to classify them in normal and session-endangering classes. If suspicious movements are identified, then the app enacts a second decision level and activates the heart or body detection sensor to check if the watch is actually still on the user's wrist. The two-level architecture tries to optimize energy consumption. To validate our system, various measurements were carried out with the aim of mapping the typical gestures of users who wear a smartwatch. The goal is therefore to be able to recognize certain movements, limit checks involving the optical sensors that are extremely energy hungry, and, thus, achieve a better battery recharge cycle.

Keywords: Continuous authentication · Smartwatch · Sensors

1 Introduction

A true experience of Ambient Intelligence does benefit from the availability of wearable devices forming a Body Area Network (BAN), both because these devices bring all needed information to the final users while moving around in the environment, and mainly because they can automatically perform some finer-grained tasks relieving users from direct intervention in boring activities. The plethora of tasks that can be imagined as being of no interest and should become completely transparent to the users can be roughly classified as harmless and risky categories; the latter ones include all those activities that are strictly tied to the identity or role of the final user. Technical evolution from the hardware side continuously shows oscillatory behavior, proposing solutions either made of many simple and single-task oriented devices or involving a few multi-purpose and more powerful devices. These two scenarios pose different problems when considering security and energy-related issues. The former scenario has a wider set of potential point-of-failure due to the number of nodes in the BAN, however, the probability of energy fault per task decreases; the latter can provide better

J. Bravo et al. (Eds.): UCAmI 2022, LNNS 594, pp. 1–12, 2023.
https://doi.org/10.1007/978-3-031-21333-5_101

support security measures but requires cleverer energy management. In order to access sensitive data or features on a device or network, a user identity first needs to be verified through an authentication mechanism. The purpose of this procedure is to protect sensitive information from any unauthorized access by malicious parties. There are different authentication techniques that depend on three types of recognition factors:

- knowledge factor, that is linked to something that only the user knows (eg. password, PIN codes, pattern lock)
- ownership factor, that is linked to something that only the user owns (e.g. electronic card, token, digital certificates)
- inherence factor, or linked to something intrinsic to the user himself (e.g. biometric parameters)

Authentication is considered strong if it takes into account at least two of the factors just mentioned: we speak of multi-factor authentication. An authentication mechanism can also be classified according to its duration. A secure solution is that of "continuous authentication". Such a mechanism occurs continuously over time. In particular, after the first authentication event and for the entire duration of the session, some parameters are monitored in order to confirm or deny the identity of the user. If the monitored values do not meet the expected values, the session is terminated. The parameters just mentioned are generally inherence factors such as biometric traits that uniquely identify who is using a device. In this way, the user only has to worry about logging in, while the subsequent part of authentication validation can be carried out in the background without requiring any active intervention. In recent years the smartwatch has undergone an exponential spread, becoming a device used by hundreds of millions of people. The features made available to the user have increased continuously, making it an increasingly useful tool in daily use. At the same time, the amount of sensitive data to manage and keep safe has also increased, such as passwords and bank details for payments via NFC. In this context, security has played a fundamental role both for the user and for smartwatch manufacturers. To contain possible vulnerabilities, various authentication methods based on passwords, PINs or pattern locks have been implemented. However, the usefulness of these techniques proved to be limited and not powerful enough to protect the user in the event of a side-channel attack and shoulder-surfing.

In this paper, a possible solution for the realization of a continuous authentication system on a smartwatch is studied. More in detail, the goal is to develop software that can monitor data from the accelerometer, gyroscope and heart rate sensor and, based on the results, decide whether the authentication session should remain open or not. The smartwatch must therefore be able to understand if there is a risk that the owner has been robbed and, in this case, it will have to automatically block the session.

2 State of the Art

Over the years, various solutions have been proposed to perform authentication and ensure continuity of use by a user. Zhang et al. [1] use behavioral biometrics

for a lightweight authentication by analyzing tapping rhythms of users. Based on the DBSCAN clustering algorithm, they perform classification first seeking core objects and then leveraging them to get the correct association with true users. In [2] a new method is proposed as a challenge-response scheme, in which the challenge is a random sequence of multiple vibration types that are already built into current smartwatches. Based on the fact that vibration is absorbed, reflected, and propagated differently according to the physical structure of each human body, the responses to vibrations are measured by the default gyroscope and accelerometer sensors in smartwatches. Lu et al. [3] have experimented with a technique capable of allowing authentication by recognizing the movements performed by a user during the phase of entering a code on the smartwatch. The algorithm developed, VeriNet, uses the data detected by the accelerometer and gyroscope to distinguish a user from a possible impostor and, at the same time, to recognize and authenticate the different passwords/pins entered by a user. In [4] the authors suggest to use data from the sensors of both the user's smartwatch and the user's smartphone to achieve identification accuracy. Guerar et al. [5] have proposed a technique, called 2GesturePIN, which allows you to authenticate yourself on the smartwatch by using, depending on the device used, the rotating crown or the side wheel. These two tools, through their rotation, are used to enter the digits of a PIN code. The system is structured so that there are two circumferences on the screen, the first corresponding to the PIN and the second relating to the rotation of the crown or wheel. The purpose of the user is therefore to identify himself by matching the two circumferences. All this is done without having to interact with the watch's touchscreen, thus foiling any attempts to steal the PIN by tracking sensor data and making shoulder surfing useless. Another similar technique is that of the CirclePIN [6], which uses a random map to associate a color to each of the digits from 0 to 9 that make up a PIN. On the screen, the colors are distributed on a circumference and the user, using the crown or the wheel, must select the right one based on his PIN. In addition to the advantages in terms of safety, these techniques however present some criticalities that limit their actual implementation. VeriNet, despite claiming to exceed existing approaches by 3–4 times, is not infallible and can therefore misinterpret some of the data processed. On the contrary, 2GesturePIN and CirclePIN, being forms of PIN entered by the user, cannot be wrong: they recognize or reject the user. Their main problem is instead linked to ease of use, limited by the fact that the user must necessarily interact with the crown or wheel to complete the login. Another technique aimed at foiling shoulder surfing attacks is described in [7] where the authors suggest to mix fake gestures inside the authentication sequence in a way that is similar to what originally suggested for smartphones in [8].

3 Our Solution

In this section we present an app for Wear OS that can guarantee continuous authentication using accelerometer, gyroscope, heartbeat , and body recognition

sensors. The accelerometer and gyroscope data are used to identify suspicious movements that may jeopardize the security of the watch. Heart sensor and body recognition instead provide, in two different ways, data relating to whether the smartwatch is still on the wrist. In particular, the latter two sensors are used to create two different versions of the same application: the first, the one with a heart sensor, safer but also more energy-intensive than the counterpart with a body recognition sensor.

In the case of the Fossil Gen 5, the accelerometer and gyroscope are sealed inside the same component, the LSM6DSO manufactured by STMicroelectronics. This system is designed to operate at low powers and is therefore ideal for devices such as smartwatches. The accelerometer is able to measure accelerations between $+/-$ 16 g, while the gyroscope detects angular velocities in the range $+/-$ 2000 rad/s.

Heart and body recognition sensors only work if the user is actually wearing the smartwatch. They use an optical sensor positioned on the back of the device and placed directly in contact with the skin. The technology used is called "photoplethysmography" and consists in the use of green and red LEDs combined with light-sensitive photodiodes, or sensors capable of measuring the wavelength of an electromagnetic wave. The operation is simple and is based on the fact that the blood reflects the red light and absorbs the green one. The optical sensor then activates the LEDs hundreds of times per second and simultaneously records the amount of reflected and absorbed light. As blood flow increases or decreases, the absorption of green light also changes, thus allowing you to count the beats. The difference between heart sensor and body recognition sensor is therefore to be found only in the period of time in which the LEDs remain on. It takes several seconds (10 to 15 s) to measure your heart rate, it takes less than one to detect your body.

3.1 Application

The purpose of the application is to protect a user's smartwatch from unwanted access for the duration of a session of use (on-off). In this process, sensors play a central role, as they provide the information necessary to understand whether the device has been stolen. At the code level, the data collection part is achieved through the use of four classes: *Sensor*, *SensorManager*, *SensorEvent* and *SensorEventListener*. Access to each sensor is managed by *SensorManager* which, through the *registerListener* and *unregisterListener* methods, allows you to activate or deactivate it. *SensorEvent* instead represents a Sensor type event and contains information such as sensor type, timestamp, accuracy and measured data. Finally, *SensorEventListener*, takes care of managing the notifications received by the *SensorManager*. The fundamental method of this class is *onSensorChanged*, which is called whenever a sensor measures a new event.

Once the application is started, the accelerometer begins to measure external events. The *onSensorChanged* method detects these changes and, based on their value, makes a decision:

- if the acceleration is within a suspicious interval, the accelerometer is turned off and the gyroscope activated for a further check;
- if the acceleration is beyond a limit threshold, the accelerometer is turned off and the heart / body detection sensor is activated.

In the first case, the gyroscope is used to understand if the data measured by the accelerometer are really suspicious. If they are, the heart sensor is activated, otherwise the gyroscope is turned off and the accelerometer is turned on again. In the second case, the detected acceleration values are so high as to assume that a theft is in progress. Consequently, the heart sensor must be activated immediately. The latter analyzes the heartbeat and, if it finds it, it turns off and the accelerometer is reactivated. If a beat is not detected the application freezes, the sensors are disabled and the session terminated. In the case of the body detection sensor, the operation is the same. The main flaw of this solution, however, lies in the fact that this sensor is unable to recognize whether the smartwatch is still on the wrist or not, but only whether or not it is in contact with the skin. This means that, if the device is removed and kept in contact with the sensor inside the hand, the application does not detect any problems.

The app is designed to work continuously throughout a session despite the presence of the Android ambient-mode system. Ambient-Mode is a mode created to reduce battery consumption by pausing an app when a user stops interacting with the smartwatch. The app starts working again only when WearOs detects the reactivation of the watch (interactive-mode). However, there are special cases of applications that can work in both ambient and interactive mode: their name is always-on app. In this project it is essential to develop an always-on app that continuously collects data from sensors. Thus it was necessary to add a code string to the app manifest as defined on the "Android Developers" portal [9].

3.2 Motion Pattern

The different sensors on a smartwatch are energy-intensive and quickly drain the battery. Hence, we developed a thresholds-based algorithm to distinguish between risky movements and simple routine gestures such as checking the display or paying. In this way, the heart and body sensors are activated only if a suspect movement is detected.

To identify routine movements we tested 3 different subjects (two wearing the watch on the left, the third on the right), each of which performed 30 repetitions for each movement. All sensors measure events every 225 ms.

Checking the Display. The smartwatch is used to check the time and, above all, to manage notifications from the associated smartphone. Controlling the screen is therefore a fundamental action, repeated several times throughout the day. The central part of this movement, that is the rotation of the wrist towards the inside of the arm, can be mapped with the use of the gyroscope alone. To understand when to turn on the gyroscope, however, it is first necessary to understand which accelerometer values are a symptom of this gesture.

Taking an average of the values collected on the 3 users, it appears that, in the initial phase of the gesture, the acceleration along x is equal to 9.0 m/s^2, along y at -7.7 m/s^2 and along z a $-$ 6.0 m/s^2. It is therefore clear how controlling the display causes precise stresses on the accelerometer. To be able to recognize them, 3 thresholds are defined, the exceeding of which can be linked to the fact that the movement has occurred. The thresholds are established on the basis of the minimum and maximum values recorded in all repetitions, with the addition of a tolerance range of 0.5 m/s^2:

– [5,6 13,7] m/s^2 on x;
– [–12,8 –5,0] m/s^2 on y;
– [–9,3 –3,5] m/s^2 on z.

Once the acceleration values beyond the previously mentioned thresholds have been detected, it is possible to move on to the study of the data produced by the gyroscope. The action of checking the time is relatively fast and takes just over half a second. Considering the sampling rate mentioned above (one sample every 225 ms), you are able to get two good measurements for each repetition of the movement. This means that, approximately, the action can be evaluated as the set of two gestures, called "phase 1" and "phase 2", which occurred one after the other. By observing the data it is possible to note that the components of the angular velocity most affected are those around the x and z axis. The data measured on y are not particularly relevant: many angular velocity values oscillate around 0 rad/s, relatively small numbers that can be caused by several small movements of the smartwatch. By evaluating the data relating to the angular velocity around the x axis, it is instead possible to notice a precise trend in all three users. The graphs show how the values measured in the first part (blue line) of the movement tend to be lower than those measured in the second (red line). Moreover, it is understood that the measured data are repeated constantly within an interval. In order To define this interval, we consider as extremes of the range, the minimum and maximum values of each phase. Starting from these numbers, establishing a tolerance interval of about 0.5 rad/sec and approximating to a digit to the right of the comma, the resulting ranges for the x axis are:

– [1,2 8,5] rad/s on phase 1;
– [1,4 9,4] rad/s on phase 2.

The last axis to consider is the z. The angular velocity measured on this axis is positive for user 1 (watch on the right wrist), negative for users 2 and 3 (watch on the left wrist). The graphs below represent the evolution of speed in absolute value in the two phases for all three users. Unlike the values measured on x, in this case there is no clear relationship between the trend of the curves. In particular, the two curves tend to often intersect between one repetition and the other, a sign of the fact that there is no phase that always predominates over the other in terms of speed. The relevant data is instead the value of these speeds. In fact, going to evaluate the averages, we note how |w| is always clearly

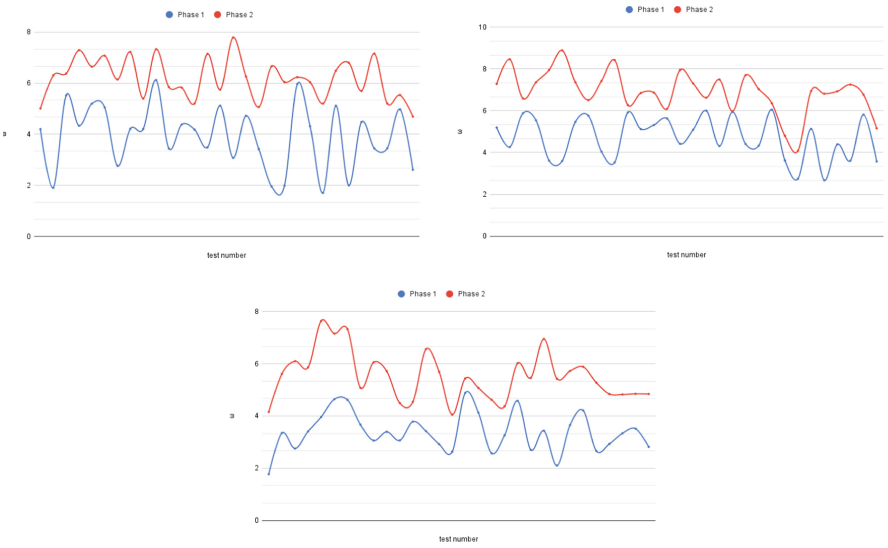

Fig. 1. |w| trend on the x axis for users 1, 2 and 3.

different from 0 rad/s. This means that the rotation of the wrist around the z axis is a gesture that occurs with certainty every time the smartwatch display is checked. As previously for the x axis, the minimum and maximum values of |w| are evaluated recorded to define a range. Also in this case a tolerance of 0.5 rad/s is used. The ranges obtained by approximating are:

- [0,9 5,3] and [–5,3 –0,9] rad/s on phase 1;
- [0,8 4,6] and [–4,6 –0,8] rad/s on phase 2.

Paying. Another very important, although less frequent, gesture is payment via NFC. NFC, also called proximity communication, is a transceiver technology that allows wireless communication. In the case of smartwatches, it is used to make payments with the simple gesture of bringing the watch close to a reader. More specifically, the action to be taken is to stretch the arm and rotate the wrist outwards, in order to bring the watch as close as possible to the reading device. Studying the data, it appears that the gesture of extending the arm causes acceleration values clearly different from 0 m/s^2 only along the x and z axes: the averages are respectively 4.4 m/s^2 and 5.8 m/s^2. By repeating the procedure already carried out in the case of the display control, 3 thresholds are established which, if exceeded, signal the possible gesture of payment:

- [1,6 7,3] m/s^2 on x;
- [–1 1] m/s^2 on y;
- [3,0 9,0] m/s^2 on z.

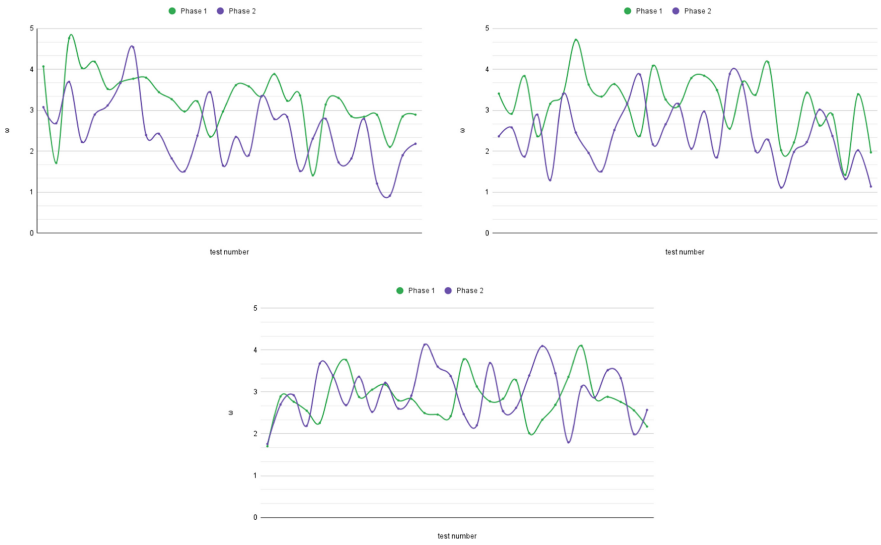

Fig. 2. |w| trend on the z axis for users 1, 2 and 3.

Turning to the evaluation of the gyroscope data, it is also clear in this case that it is not possible to consider the data relating to the y axis as they are too variable from user to user.

By evaluating the graphs below relating to x, we note instead how the smartwatch is always stressed around this axis: the |w| they are in fact constantly greater than 0 rad/s. From these data, however, there is no clear trend in the relations between the two phases: the |w| average is greater in phase 1 for users 1 and 3, the opposite for user 2. This can be translated into the fact that the values recorded on x are influenced by the type of user making the movement. The only certainty that can therefore be established is that to pay it is necessary to rotate the wrist and then apply a $|w| > 0$ around the x axis. By repeating the procedure implemented in the case of display control, you can search for ranges within which you are able to identify the gesture. To do this, the graphs relating to the highs and lows of the two phases are still used. Approximating, applying a tolerance of 0.5 rad/sec and considering that the rotation around x is negative, the resulting intervals are as follows:

- [–7,5 –0,6] rad/s on phase 1;
- [–6,6 –0,5] rad/s on phase 2.

For the z axis, the situation changes: it is immediately evident how the curve of phases 1 (green) always stays above the curve of phases 2 (purple) for all users. The trend is therefore repetitive and, regardless of the user considered, |w| in the initial part of the movement is always greater than that in the final part. Approximating and establishing a tolerance of 0.5 rad/s, the ranges are therefore:

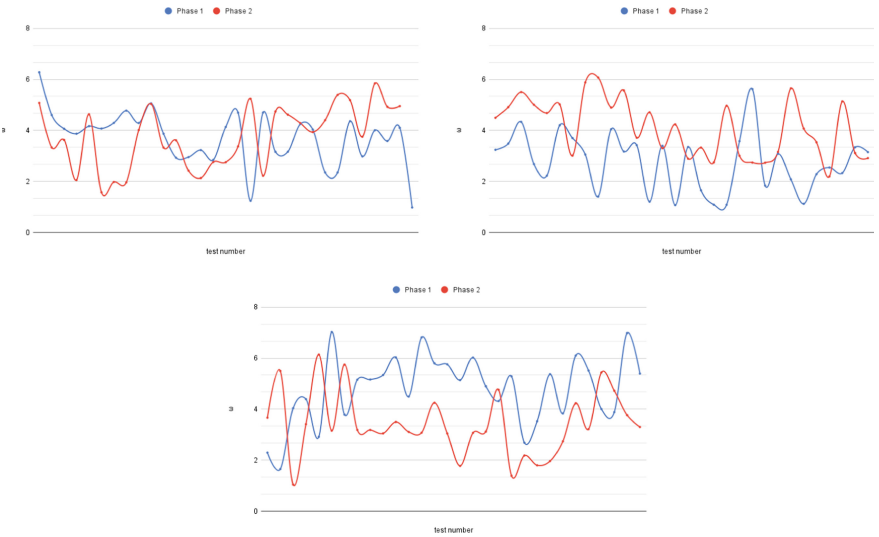

Fig. 3. |w| trend on the x axis for users 1, 2 and 3.

– [0,9 5,4] and [–5,4 –0,9] rad/s on phase 1;
– [0 4,8] and [–4,8 0] rad/s on phase 2.

Movements at Risk. Up to now, the movements considered safe have been described, for which it is not necessary to carry out the control by means of a heart sensor. As already mentioned, however, there are also risky ones, considered as such because they put the security of the smartwatch at risk. All these movements can be traced back to a maxi-group, that of thefts. In fact, considering that a smartwatch is always on the owner's wrist, this is the only way to get hold of it. In the case of thefts carried out with the use of force, the modus operandi of a criminal is to grab the smartwatch and pull with enough force to break the strap that keeps it tied to the wrist. The action, which must be rapid and impetuous, is characterized by the fact that the smartwatch undergoes strong jolts in one or more directions. For this reason, the choice of the sensor to be used to detect these movements fell on the accelerometer. The measurements made have shown that, approximately, the alarm threshold can be set for acceleration values higher than 20 m/s^2 along at least one of the three axes. This uncertainty is due to the fact that, to find data close to reality, it is necessary to apply such force as to damage the watch. As it was not possible to do this, a value was chosen which, while not breaking the strap, is the result of a strong tug on the wrist. This solution can still be considered reliable, given that the accelerations that a smartwatch normally undergoes are rarely above 20 m/s^2. To confirm this, tests have been carried out on a user with the aim of detecting how many times this threshold is reached in the course of a day. The data collected showed that this occurs on average no more than 2 times. Mea-

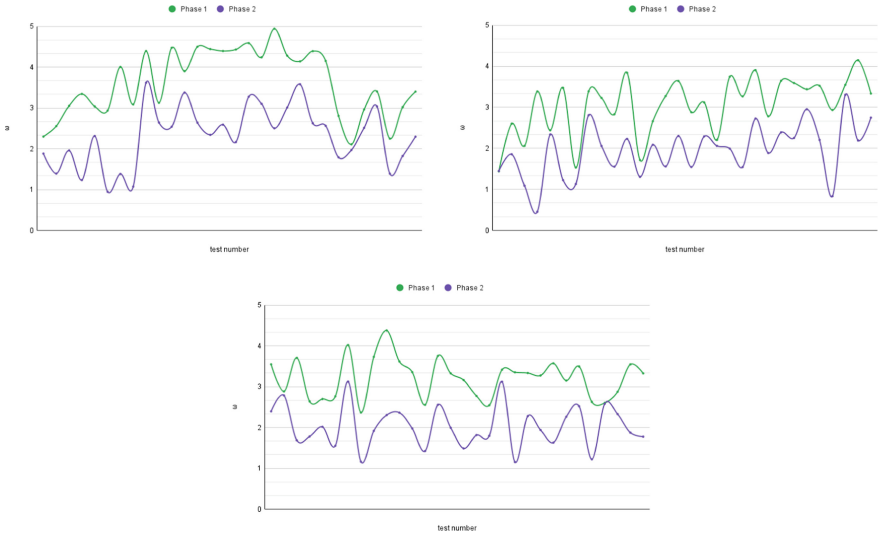

Fig. 4. |w| trend on the z axis for users 1, 2 and 3.

suring values over 20 m/s^2 therefore means that you are already sure that some suspicious movement is taking place for which it is worth activating the heart or body recognition sensor. Furthermore, given the low probability of exceeding this threshold with normal gestures, it is possible to avoid unnecessary energy consumption due to the activation of the optical sensors.

4 Experiments

The biggest challenge in this project is to reconcile the continuous use of sensors with acceptable energy consumption. As seen above, the developed app uses the accelerometer most of the time, activating the other sensors only in the presence of particular triggers. In this way it is possible to significantly reduce the activation time of the gyroscope and optical sensors, with consequent energy savings. To quantify these consumptions, tests were carried out on the same users used to map the movements. Each user wore the smartwatch for 3 different days, with active, in addition to sensors, bluetooth and NFC.

The tests are intended to try to understand if the watch is able to stay on for a whole day with the app active.

Heart Sensor. As shown in Table 1, the average for all three users is 371 min of runtime, 2 suspicious movements were detected.

Table 1. Heart sensor

	Day 1	Day 2	Day 3	Average	Battery life
User1	0.277 %/min	0.278 %/min	0.293 %/min	0.282%/min	355 min
User2	0.266 %/min	0.272 %/min	0.281 %/min	0.273%/min	366 min
User3	0.254 %/min	0.258 %/min	0.250 %/min	0.254%/min	394 min

Body Recognition Sensor. As shown in table 2, the average for all three users is 352 min of runtime with 2 suspicious movements detected.

Table 2. Body recognition sensor

	Day1	Day2	Day3	Average	Battery Life
User1	0.309 %/min	0.304 %/min	0.267 %/min	0.293%/min	341 min
User2	0.284 %/min	0.288 %/min	0.281 %/min	0.284%/min	352 min
User3	0.265 %/min	0.284 %/min	0.279 %/min	0.276%/min	352 min

Discussion. The optical sensors are activated a few times during a day: this means that the app is able to classify as routine almost all the movements performed with the exception of a few sudden movements. By studying the estimated battery life values, it can be seen that the choice to use the heart sensor or the body recognition sensor does not significantly impact the performance of the app: the smartwatch lasts about 6 h in both cases. The variance between the measurements of the same user or between different users, all other things being equal, are mostly due to external lighting. In fact, the smartwatch tends to increase the brightness of the screen in relation to how much light there is in the place where you are. This behavior causes higher consumption in the case of a user who uses the watch outdoors than one who spends more time indoors. The values higher than the average measured for each user can therefore be explained by this reasoning. For the same reason, the average duration in the case of the body recognition sensor (352 min) is lower than that of the heart sensor (371 min): the days in which the measurements of the first case were carried out were in fact much sunnier than the second case.

5 Conclusions

The purpose of this paper was to create a continuous authentication system for smartwatches based on the detection of suspicious movements through motion and optical sensors. The developed app has proved to be valid in identifying the majority of movements like violent thefts and other risky situation. On the energy consumption side, the app has proved to be particularly energy-intensive due to the continuous use of sensors. Although their use has been optimized in

such a way to impact the battery as little as possible, the duration of a charging cycle has been estimated around 6 h. These numbers testify that the app, even if it works, cannot currently be considered for a real application. Any user expects the smartwatch to stay on from morning to evening, which is currently impossible with the authentication app running. A session of use should be at least 12 h, almost double the autonomy found on the Fossil Gen 5. Using other devices, battery life is likely to increase, but still not enough to meet the needs of an average user. Currently, the only hope for these types of applications that highly exploit sensors is that the smartwatch market will be able to offer increasingly high-performance models both in terms of battery life and optimization of the operating system. Before that moment it is very complicated to be able to use solutions such as the one proposed in this paper.

References

1. Zhang, H., Xiao, X., Ni, S., Dou, C., Zhou, W., Xia, S.: Smartwatch user authentication by sensing tapping rhythms and using one-class DBSCAN. In: Sensors, vol. 21, no. 7 (2021)
2. Lee, S., Choi, W., HoonLee, D.: Usable user authentication on a smartwatch using vibration. In: Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security, pp. 304-319 (2021)
3. Lu, C.X., et al.: VeriNet: user verification on smartwatches via behavior biometrics. In: Proceedings of the First ACM Workshop on Mobile Crowdsensing Systems and Applications (2017)
4. Weiss, G.M., Yoneda, K., Hayajneh, T.: Smartphone and smartwatch-based biometrics using activities of daily living. IEEE Access **7**, 133190–133202 (2019). https://doi.org/10.1109/ACCESS.2019.2940729
5. Guerar, M., Verderame, L., Migliardi, M., Merlo, A.: 2GesturePIN: securing pin-based authentication on smartwatches. In: Proceedings of the 28th IEEE International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises, 12-14 June 2019, Capri (Napoli), Italy
6. Guerar, M., Verderame, L., Merlo, A., Palmieri, F., Migliardi, M., Vallerini, L.: CirclePIN: a novel authentication mechanism for smartwatches to prevent unauthorized access to IoT devices. ACM Trans. Cyber-Phys. Syst. 4(3), 19 (2020)
7. Park, M., Aburada, K., Okazaki, N.: Proposal and evaluation of a gesture authentication method with peep resistance for smartwatches. In: Ninth International Symposium on Computing and Networking Workshops (CANDARW) **2021**, 359–364 (2021). https://doi.org/10.1109/CANDARW53999.2021.00067
8. Guerar, M., Migliardi, M., Merlo, A., Benmohammed, M., Palmieri, F., Castiglione, A.: Using screen brightness to improve security in mobile social network access. In: IEEE Transactions on Dependable and Secure Computing, vol. 15, no. 4, pp. 621-632, 1 July-Aug 2018. https://doi.org/10.1109/TDSC.2016.2601603.
9. Keep the device awake. https://developer.android.com/training/scheduling/wakelock