Invisible CAPPCHA: A Usable Mechanism to Distinguish Between Malware and Humans on the mobile IoT

Meriem Guerar, Alessio Merlo, Mauro Migliardi, Francesco Palmieri

PII: S0167-4048(18)30755-7
DOI: 10.1016/j.cose.2018.06.007
Reference: COSE 1363

To appear in: *Computers & Security*

Received date: 27 February 2018
Revised date: 19 June 2018
Accepted date: 21 June 2018

Highlights

- Movement-driven invisible CAPTCHA for smartphones
- Overcome the limitations of CAPTCHAs without requiring user input
- Reliably and transparently identify the presence of a human performing an operation
- Leverages micro-movements generated naturally by interactions with touch-screen
- Prevent automated programs from abusing cloud services from mobile devices

Invisible CAPPCHA: A Usable Mechanism to Distinguish Between Malware and Humans on the mobile IoT

Meriem Guerar[c], Alessio Merlo[b], Mauro Migliardi[c], Francesco Palmieri[d]

[a]*DEI, University of Padova, Italy*
guerar@dei.unipd.it
[b]*DIBRIS, University of Genoa, Italy*
alessio@dibris.unige.it
[c]*DEI, University of Padua, Italy*
mauro.migliardi@unipd.it
[d]*Department of Computer Science, University of Salerno, Italy.*
fpalmieri@unisa.it

Abstract

Smartphone devices are often assuming the role of edge systems in mobile IoT scenarios and the access to cloud-based services through smartphones, for transmitting multiple sensory data related to human activities, often implying some lawful evidence, has become increasingly common. Thus the need for protecting such transactions from abuses and frauds based on automation techniques is now a critical issue. The most widely adopted method to prevent unauthorized access and abuse of a service by malicious software automation is CAPTCHA. However, trying to strengthen CAPTCHA resilience to automated attacks has led to challenges that, while still being vulnerable, are both difficult and unpleasant for humans. Hence, the strong need for a mechanism that is both secure and usable. In this paper, we present Invisible CAPPCHA, a mechanism that, leveraging trusted sensors embedded in a secure element located on a smartphone is capable of separating humans from computers in a way that is completely transparent to users. Furthermore, as no challenge is required, no additional time is needed and the user cannot fail it by mistake. Compared to the state of the art, our proposal is both secure and more user friendly, lending itself optimally to secure mobile cloud services.

Keywords: Smartphone, IoT, Usable Security, Automatic Fraud Detection and Prevention, CAPTCHA, Invisible CAPPCHA

1. Introduction

Smartphones have become the main mean through which users can access cloud services to go shopping, send messages, reserve tickets for events or seats at a restaurant, post comments into a blog, etc. These devices are also assuming the role of edge systems in mobile IoT scenarios, by collecting and pre-processing the data generated by multiple sensors, such as the ones used in healthcare monitoring or tracking/surveillance services and conveying them towards the cloud for further and deeper analyses. However, it is known that most of these services can be abused and are subject to automated attacks generated by computer programs called bots, or exploited by automated applications often mimicking the human behavior for fraud or criminal purposes. As an example of abuse, we may cite the recent trend in posting inflammatory comments on social media to apply pressure on citizens during election campaigns [1]. Furthermore, malicious programs may also engage in damaging or manipulative behaviors such as register thousands of free accounts at a time, vote automatically in online polls, click on ads to generate revenue or to reduce the likelihood of being displayed to a real user, iterate through the entire space of passwords to find the missing credential part, impersonate humans in their activities, by tricking remote workers' monitoring or activity tracking systems as well as creating the illusion of a physical presence in some place for creating a false alibi.

In a Content Delivery Network (CDN) scenario where edge devices are adopted to dynamically cache contents on the basis of users' requests, automated access to unpopular contents might degrade the system performance and, if mitigating mechanisms are not implemented, the degradation might scale to complete saturation of bandwidth and storage space. Furthermore, as it is nowadays easier for a smartphone to be always on than it is for a traditional PC, mobile botnets [2, 3] are an emerging threat and are starting to appear in the wild. In such a scenario, the capability to throttle access to critical mobile cloud services controlling that the request has been activated by an actual human user would mitigate the level of threat and reduce the potential of mobile cloud services abuse. While the machine-to-machine IoT has a different set of security priorities [4], in the smartphone-empowered IoT it becomes necessary to reliably demonstrate the human nature of a smartphone user before connecting to some online service or when transmitting to a cloud application some sensory data such as a GPS position or a real-time video as well as an ECG track. The easiest and most popular way used over years by the web developers to tackle this issue is to allow access to these services only to users able to solve a CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart). The main purpose of CAPTCHAs is, as explained by the name itself, to distinguish between humans and computer programs; this goal is pursued by generating a test that is intended to be easy to solve for humans, yet hard to solve for computers. The most widely used CAPTCHA is reCAPTCHA [5] by Google which has been adopted by many popular websites.

Over the years, reCAPTCHA has leveraged text-, image- or audio-based challenges to verify the presence of a human before allowing access to online services. The original reCAPTCHA challenge asked users to recognize a distorted text in an image form which is known to be difficult for computer vision systems but easy for humans' eyes. However, the increasing sophistication of computers and artificial vision systems forced reCAPTCHA's challenges to become more complicated by adding noise and distortion aimed at making it harder to break. Nevertheless, it has been solved by automated systems with high level of accuracy, e.g., [6, 7, 8]. In addition, the introduction of heavy noise and distortion makes the challenge, especially on small screens such as the ones of smartphones, hard for humans to decipher as well. In 2014, Google began to replace the reCAPTCHA system with a user-friendly alternative called No CAPTCHA ReCAPTCHA [9]. This version requires simply the user to click in "I'm not a robot" widget and through an advanced risk analysis software that relies on behavioral cues, such as where users click, how long they linger over a checkbox, their typing cadence and other variables that Google is keeping secret, tries to determine which behaviors are human-like and which are too robotic. If the system classifies the user as human, then it gives him access. However, when the risk analysis engine cannot confidently predict whether a user is a human or not, it falls back to prompting the user to solve a reCAPTCHA challenge. Similar to No CAPTCHA ReCAPTCHA, the latest version, the Invisible reCAPTCHA [10] employs an advanced risk analysis software to determine if a visitor is a human or not but without any direct involvement of the users themselves. In this version, the users are not required to click in "I'm not a robot" widget unlike the previous one.

To overcome the limitations of CAPTCHAs, in a past work we have introduced the concept of Completely Automated Public Physical test to tell

Computer and Humans Apart (CAPPCHA) [11, 12], which, in order to spot the presence of automations, leverages the physical nature of human subjects instead of requiring them to tackle a complex cognitive task. In doing this, CAPPCHA asks the user to tilt the smartphone to a specific degree to be recognized as a human. CAPPCHA is designed in such a way that users cannot make mistakes and there is no additional cognitive burden places onto them; yet, just like traditional CAPTCHAS, it introduces a specific additional step in the chain of tasks that lead to the user's goal.

Thus, to further reduce the perceivable impact on end-user, while guaranteeing the same level of reliability in identifying the presence of a human performing an operation, we introduce a new version of CAPPCHA, Invisible CAPPCHA, whose goal is to be fully transparent to the user without requiring any additional task or challenge to prove that he/she is human.

In conceiving such idea, we noticed that almost all the online services that require protection against automation abuses require user's input (e.g., fill a form, write a comment, tap on a button, perform login/ sign up, etc.). While on computers the user's input is limited to the keyboard and mouse, the user's input on smartphones leverages the touch paradigm and makes heavy use of tapping gestures; as a consequence, the user's tap on smartphone is a physical interaction that is naturally used to fill forms, write comments, download an application, acknowledge a message etc. This interaction generates a micro-movement of the device that can be easily detected and measured by sensors such as the accelerometer that are universally available on modern smartphones.

Thus, instead of asking the user to perform an additional motion task (e.g., tilting the smartphone) to authorize access to the requesting application, Invisible CAPPCHA leverages the micro-movements of the device which are generated naturally by the user's interaction with the touch-screen, more in details the taps. These micro-movements need to be measured by a trusted sensor to prevent sensor data manipulation by any malware component. As all the cloud-based services that need confirmation that they are dealing with human requires user's input, Invisible CAPPCHA can be used as an effective and secure way to prevent automated programs from abusing cloud services from mobile devices; furthermore, this result can be achieved without posing any additional burden on the user, mainly in services that require periodic transmission of sensory data, and thus having an extreme level of usability. Such proposal, allowing smartphones to assume the role of reliable edge gateways in human-centric IoT/cyber-physical applications, has been carefully assessed from the functional point of view in order to provide a proof of concept for all the involved ideas and mechanisms. The results demonstrated the effectiveness of the whole approach and architecture, and its real applicability in both present and future devices.

2. Backgrounds and related literature

2.1. Besides CAPTCHAs

The most widely-deployed form of CAPTCHA is text based, where distorted texts are shown as CAPTCHA images. However, in addition to the usability issue, research has shown that today's Artificial Intelligence technology has become sophisticated enough to solve the hardest challenges with 99.8% accuracy, e.g., [6]. To address this issue, Google introduced the aforementioned "No CAPTCHA reCAPTCHA" system, based on an advanced risk analysis engine to consider how users interact with CAPTCHA verifications. If the risk analysis engine determines that the user is human, the user is only required to tick the "I'm not a robot" checkbox and be verified without needing to solve a CAPTCHA. Otherwise, the user will be presented with an image-based challenge or a traditional text-based CAPTCHA to verify the human nature of the user. However, Sivakorn et al. [13] show how Google tracking cookies can be used to fool the risk analysis system into thinking that a program was a human, and to check the "I'm not a robot" box. In addition, in [14], authors claimed that a CAPTCHA challenge will always be required if the Google web cookies are deleted, an incognito web browser session is used, or JavaScript is disabled. Hence, if these actions are performed by a malware or by the legitimate user, the main motivation of using "No CAPTCHA" (i.e., optimize the user experience by allowing them in many cases to skip the CAPTCHA tests entirely) will be ineffective. Recently, Google improved this mechanism and made it entirely invisible. The new "Invisible reCAPTCHA" service [10] is based on the same technology as the "No CAPTCHA reCAPTCHA", but it removes the check-box step, while suspicious users and bots will still have to deal with reCAPTCHA's various challenges.

2.2. CAPTCHA strengthened authentication and its alternatives

In the literature, CAPTCHA has been also used to improve the security of user authentication methods. For instance, Pinkas et al. [15] proposed to combine the password with CAPTCHA to counter online dictionary attacks. In [16, 17] , authors suggested to use CAPTCHA in graphical password schemes to resist spyware attacks. Yeh et al. [18] introduced a mobile user authentication system in cloud environments that uses CAPTCHA to protect cloud servers against malicious registrations and logins. Pequegnot et al. [19] suggested to use CAPTCHA to improve the security of PIN codes on mobile devices against automated attacks. Recently, Althamary el al. [20] proposed a CAPTCHA based authentication method in cloud environments to strengthen weak passwords against different attacks including short-password attack, dictionary attack, keyloggers, phishing and social engineering. However, the CAPTCHA schemes used in these authentication methods are similar to existing commercial text-based CAPTCHAs and all of them have been broken with high percentages of accuracy, e.g., [6, 21, 22, 23] and [24]. Besides security issues, it is common knowledge that visual CAPTCHA such as the text-based schemes do not properly fit the smartphone form factor [25]. These facts motivate many researchers to design new ways for preventing automated attacks suitable for mobile devices.

Guerar et al. introduced CAPPCHA [11], a new dependable way to determine whether a user is human. The CAPPCHA challenge requires the user to tilt the smartphone to a specific degree displayed on the screen to be recognized as a human. The idea behind this, is that malware can affect the behavior and the security of the smartphone in several way (e.g., battery [26]) but it cannot physically move the device. The movement detection of the smartphone is measured by an accelerometer sensor embedded in the secure element to prevent sensor data manipulation by the malware. Beside acting as standard CAPTCHA, CAPPCHA can be used to enhance the security of PIN authentication against mobile malware. In [12], authors presented an extended usability study of CAPPCHA based on 200 volunteers. Their experiment results show that CAPPCHA is easy to understand and use, and it shows a high acceptance rate by the users. Shrestha et al. [27] proposed to use the hand waving gesture to prevent unauthorized access to sensitive services. Their system uses the light and the accelerometer sensor for detecting the gesture instead of the proximity sensor. However, authors assume that the OS kernel is completely immune to any tampering and the sensor data cannot be manipulated by the malware. Guerar et al. presented BrightPass [28], an authentication mechanism that leverages the screen brightness as a communication channel that is inaccessible to the mobile malware for improving the security in mobile social network access. For each authentication session, BrightPass displays an alternating circle's brightness on the phone screen to tell the user when to input the correct PIN digit and when to input a fake one. This way, it prevents malware from successfully replaying the user's PIN, thereby disallowing the possibility to gain unwanted access and/or perform specific actions without the user's awareness. It also ensures a short authentication time (i.e., 6.73 sec) and low error rates (i.e., 1.81%). All of these solutions, however, require a specific additional activity to be performed by the user.

2.3. Smartphone built-in sensors and security

The current generation smartphones are equipped with a multitude of sensors in the pursue of making them even smarter. These sensors, on the other hand, have been exploited both for improving user's security (e.g., extract biometric features or gesture recognition for authenticating the user, etc) and for compromising it (e.g., steal user's credential, violate his privacy, etc.).

Conti et al. [29] presented a transparent method that uses data measured by the accelerometer and orientation sensor during call placing/answering as a biometric measure to authenticate the user and thus, prevent unauthorized users to perform this action. Buriro et al. [30] proposed motion based a touch-typing biometrics method to improve the security of an 8-digit PIN/password used for mobile banking applications. Their method authenticates the legitimate user based on the timing differences in the entered keystrokes and the phone-movements during the PIN/password entry process which is measured from different 3-dimensional sensors (i.e the accelerometer, the orientation, the gravity sensor, the magnetometer and the gyroscope). De Luca et al. [31] introduced a transparent authentication method to enhance the security of the Android Pattern Lock. While performing the wipe gesture to draw the pattern, some biometric attributes are collected, including XY-coordinates, pressure, size, time and speed of the touch to validate the entered pattern. Thus, the smartphone is unlocked only if the user draws the correct pattern and the way it has been drawn matches the stored attributes. In contrast to these methods that use motion-sensor readings to extract unique behavioral characteristics of individuals to distinguish between a legitimate user and an impostor, our proposed mechanism, Invisible CAPPCHA, uses motion sensor readings to distinguish between humans and malware.

Much work has been dedicated to showing how smartphone sensors can be used as a side channel to infer user's keystrokes typed on the touchscreen. The idea behind this attack is that the device's micro-movements caused by the user's tap on the touchscreen are quite different depending on tap location. TouchLogger by Chai et al. [32] is the first work that suggests to use motion sensors as a side channel to infer keys typed on a number-only soft keyboard on a smartphone. Xu et al., introduce TapLogger [33], a trojan application that uses the data collected by the accelerometer to detect the occurrence of taps and data from the orientation sensor to infer the positions of these taps. This Trojan is able to stealthily log the screen lock password and the numbers entered during a phone call, such as credit card and PIN numbers. Miluzzo et al. introduce TapPrints [34], a framework that infers the tap information on the soft keyboard of both smartphones and tablets based on accelerometer and gyroscope readings. In [35], Owusu et al. show that the data acquired from the accelerometer is sufficient to infer entire sequences of the text typed on the soft keyboard. Similarly, Aviv et al. [36] show that the accelerometer readings can be used as a side channel to infer both user's PINs and lock patterns. Recently, Mehrnezhad et al. [37] demonstrated how JavaScript access to the accelerometer and gyroscope readings can be used as a side channel to infer user's PINs. Similarly to the above-mentioned works, Invisible CAPPCHA leverages the micro-movement of the device caused by the user's tap on the touchscreen. However, our goal is to improve security instead of compromising it.

3. The Invisible CAPPCHA concept

We already discussed how the most common method to prevent automated access to remote cloud applications, web resources or sensitive mobile services is to perform some kind of test to check that a user is a human. For this purpose, Invisible CAPPCHA leverages the micro-movements of the device which are generated naturally by the user's interaction with the touchscreen. We define micro-movement a movement that, while being usually too small to be noticed by the human user and being not a desired effect of the user action, can still be measured by monitoring the acceleration of the device along the direction perpendicular to the touchscreen in order to detect both gentle and strong taps. Strong taps present variation peaks with more amplitude than the gentle taps. Similar to CAPPCHA [11, 12], the rationale behind our work is that malware cannot physically move the device. However, Invisible CAPPCHA leverages that idea in a fully transparent way that requires no additional task or challenge to be performed or tackled by the user; this is a major difference from CAPPCHA [11, 12], which asks the user to tilt the device to a specific degree as a challenge. As the phone movement caused by user's finger tap is small, compared to tilting the smartphone action, we conducted several experiments to prove that the micro-movements generated by the user tapping actions can be identified with a high level of accuracy and, at the same time, that such activity cannot be simulated by malware leveraging unsecured hardware such as the smartphone's vibration motor. In addition, in order to prevent sensor data manipulation by malware applications, the micro-movements of the device need to be measured by a motion sensor that is embedded in the secure element which is a tamper-resistant device. Fortunately, a SIM card equipped with a motion sensor already exists in the market under the name SIMSense [38]. Vivo was the first mobile network operator that introduced SIMSense to its customers in Brazil [38]. Such SIM card will allow Invisible CAPPCHA to be implemented in any existing devices. When the user fills a form or provides other information to a cloud application/service through a browser or a web service interface, the secure element checks if during the tap gesture events a tap micro-movements pattern is recognized in sensor data measured by the embedded accelerometer, typically used for measuring movements and orientation. If this is the case, then the input is considered as a valid, human-generated one, otherwise the input is considered as malicious one injected by the malware (See Figure 1).

In any case, a message accompanying the sending of some data, that has to be verified as human-generated, to the server, is used for notifying the results of the aforementioned check, so that the server is able to differentiate its behavior according to the genuine nature of human-submitted data or its generation by a malicious automation (malware). For example, an HTTP transaction based on the POST method may integrate the invisible CAPPCHA mechanism, by conveying the result message together with the other transmitted data. The integrity of such message strictly depends on the use of the secure element for managing the whole mechanism.

Since the secure element can be equipped with a digital certificate, we can strongly guarantee the integrity of the sent message as well as its authenticity, by associating it with the identity of the mobile terminal that can be strongly checked and verified. More precisely, the secure element signs the results of the verification by using ECDSA, standardized in FIPS 186-4 [39], and sends

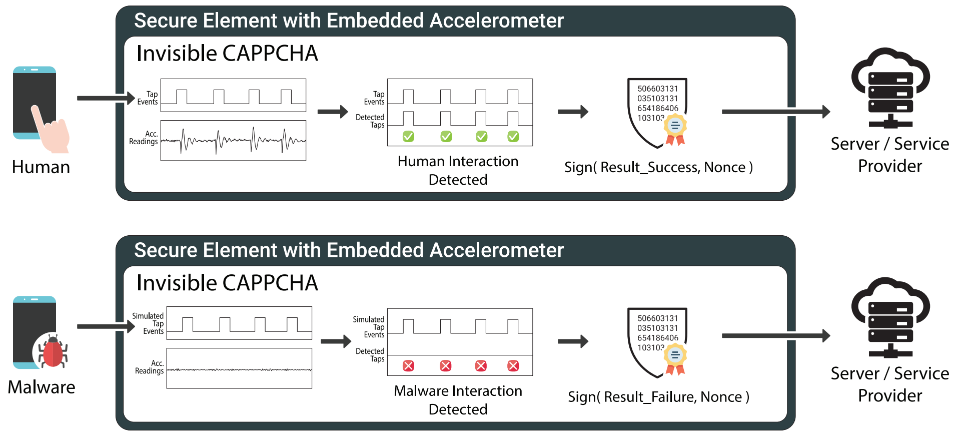

Secure Element with Embedded Accelerometer
Invisible CAPPCHA
Human
Tap Events
Acc. Readings
Detected Taps
Human Interaction Detected
Sign(Result_Success, Nonce)
Server / Service Provider
Secure Element with Embedded Accelerometer
Invisible CAPPCHA
Malware
Simulated Tap Events
Acc. Readings
Detected Taps
Malware Interaction Detected
Sign(Result_Failure, Nonce)
Server / Service Provider

Figure 1: Invisible CAPPCHA Mechanism.

it to the cloud application (server), so that any malicious tampering of the message will be immediately detectable. Such choice for the signature algorithm is motivated by the fact that ECDSA provides the same security degree as the more traditional RSA but by adopting much smaller keys. In our specific application, reducing the key size brings significant advantages, starting from the fact that the use of smaller keys implies faster algorithms for signature generation since smaller numbers are involved in mathematical operations. Reducing the size of public keys also means using smaller certificates and hence less data has to be exchanged when establishing secure connections. This has the immediate effect of containing the connection setup times and the load latency on web accesses. Furthermore, breaking an ECDSA key implies efficiently solving the Elliptic Curve Discrete Logarithm Problem on which the whole mathematical community has not made major progresses since it was introduced around 1985 by Koblitz [40] and Miller [41]. For these reasons, ECDSA is extensively used in the smartphone arena and in particular on the Apple ecosystem, both for signing messages in iMessage and syncing relies on iCloud keychain.

In our security model, the Invisible CAPPCHA mechanism is used also to enhance the security of Password-based authentication methods (i.e. used to protect access to the secure element), yet this is not necessary in smartphones equipped with the Trusted Execution Environment (TEE) [42].

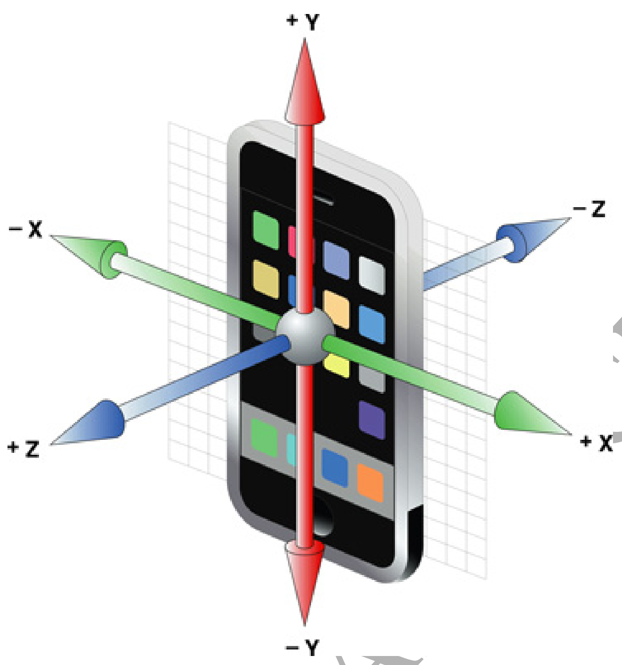


Figure 2: Accelerometer axes layout [43].

3.1. Tap detection

The accelerometer is able to sense any acceleration event interesting the smartphone over the three axes x, y and z whose directions are predefined as shown in Figure 2. The raw data that can be obtained from the accelerometer report the acceleration measured on each axis in g-force units, represented as a time series of vectors:

$$\{A_i\}_{1=1}^{n} = \{(a_1^x, a_1^y, a_1^z), \ldots, (a_n^x, a_n^y, a_n^z)\} \tag{1}$$

so that, ideally, an accelerometer embedded within a device lying on a perfectly flat surface should return an infinite series of values $\{(0, 0, \pm 1), \ldots\}$. Clearly, since accelerometer devices are not perfectly accurate, this is not true and the values observed will not be constant, but rather change on each observation due to the effect of noise; however, as our experiments clearly show, such noise is never critical to the detection of taps, hence, even in the real case of not ideal accelerometers, the detection of taps is very accurate.

Tap detection by the accelerometer is not new, and many researchers have suggested algorithms to detect tap for different purposes. For instance, Heo et al. [44] suggested an algorithm that utilizes built-in accelerometer data to distinguish between gentle and strong taps and they had proved its feasibility in various conditions (i.e., single-handed, two-handed, immersive and walking condition). The tap classification is done by calculating the sum of the absolute values of all accelerometer samples within a time window around the touch event and compare it with a predefined threshold. Davarci et al. [45] suggested to use accelerometer data to detect tap as well as determining if it belongs to a child or an adult. Xu et al. [33] developed a trojan application that detects the occurrences of tap events by monitoring the motion change caused by the external force applied on the smartphone. Beside academic work, there are currently in the market accelerometers by NXP [46] that integrate single/double and directional tap detection algorithms. The embedded algorithms for single and double tap analyze the acceleration patterns of finger taps along the z-axis by comparing thresholds and timing conditions. Similarly to these embedded algorithms, we also analyse the acceleration readings along the z-axis.

In order to implement Invisible CAPPCHA any tap detection algorithm from the literature can be used e.g.,[46]. In fact, in this paper we developed a simple algorithm which is based on two tap acceleration pattern features but we did not tested in different conditions because our main goal here is not to prove the feasibility of tap detection using accelerometer data but instead to prove that the user's tap acceleration pattern is distinguishable from patterns generated using unsecured hardware such as the vibration motor whose API can be exploited by malware. Therefore, the micro-movement caused by user's tap can be used to distinguish between human and malware. To this end, in following section we compare between tap and vibration patterns to show the difference.

3.2. Distinguishing between tap and vibration patterns

The vibration motor available in any smartphone could be exploited by malicious applications to produce a micro-movement of the device within an automation. Hence we need to show the difference between the micro-movement caused by real taps and those caused by vibration. For this reason, we compared between the accelerometer data collected during the user's tap and vibration from different smartphones, namely HTC DESIRE, GALAXY S ADVANCE, LG G4, OPPO F1 and ONEPLUS 5T. These devices represent a significant timespan as they are smartphones commercialized respectively

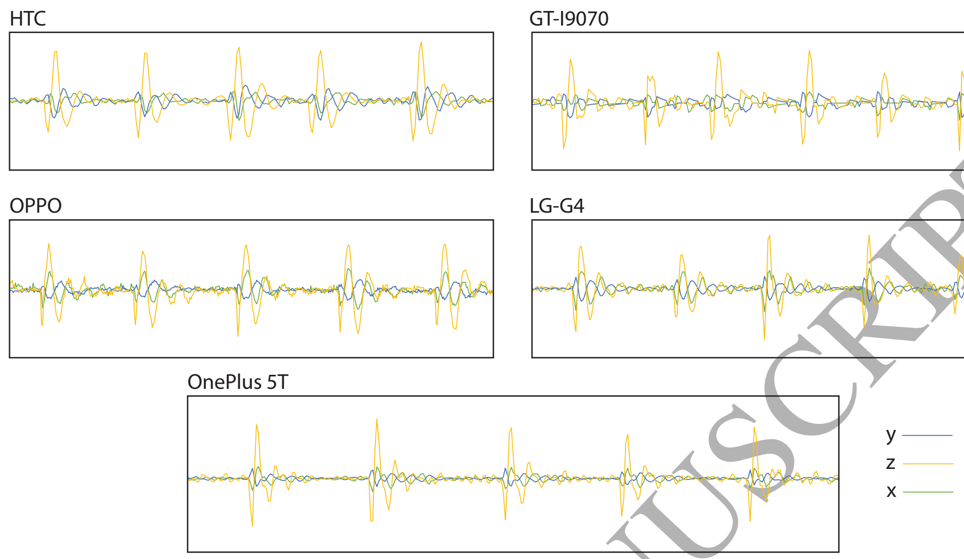


Figure 3: Acceleration readings during user's input in HTC DESIRE, GALAXY S ADVANCE, OPPO F1, LG G4, ONEPLUS 5T respectively.

in 2010, 2012, 2015, 2016 and 2017. Furthermore, they have also different types of vibrating motors.

In order to show the tap effect on acceleration pattern, we developed an Android application with a personalized virtual keypad. While the application is running, the accelerometer readings in x, y and z directions and tap event information are stored in the internal memory of the smartphone. The tap events are identified by timestamps of the received event *Motion.Event.ACTION_DOWN* triggered when the user taps his finger on the touchscreen and the event *Motion.Event.ACTION_UP* triggered when the user lifts up his finger. When gathering sensor data from the smartphone's accelerometer, it is fundamental to carefully choose a sampling frequency, whose value is able to significantly impact the accuracy of the resulting data. Clearly the best choice has to be experimentally determined through multiple trials. We used a sampling frequency *SENSOR_DELAY_GAME* in LG G4 and ONEPLUS 5T and *SENSOR_DELAY_FASTEST* in the other devices [47], which is the fastest rate at which sensor data is provided. In the experiments, we consider a scenario in which the user holds smartphone by one hand and tap on the touchscreen with the index finger of the other hand. We collected sensor data of 200 taps from the above-mentioned smartphones and repeated the experiments with several users characterized by different sex and age. As shown in Figure 3, the effect of a finger tap on acceleration change pattern is significantly higher and similar on the Z-axis in all tested devices. It goes down first, then jump up dramatically then go down again to finally ripple and settle to the beginning position.

On the other hand, we developed another application that calls the vibration API when a touch event is simulated to show the vibration effect on acceleration pattern. We simulated 20 touch events and thus we collected accelerometer data of 20 vibrations from each smartphone. As shown in Figure 4, the effect of smartphone built-in vibration motor is significantly higher on one or two axis and it differs from a device to other except ONEPLUS 5T. For example, the vibration motor in GALAXY S ADVANCE produces vibration only on the Z axis, the HTC DESIRE in the YZ axis while the OPPO F1 and LG G4 produces vibration in the XY axis. Regarding ONEPLUS 5T, during the test we noticed that its vibration is weak in comparison to the other devices and this explains why there is not a significant change on acceleration pattern in any axes.

To distinguish the micro-movement caused by real taps and those caused by vibration, we developed a simple algorithm that relies on two acceleration pattern tap features, a negative peak and positive peak which are detected by comparing acceleration readings on Z axis against predefined thresholds. The results of the implementation of this algorithm on the collected data from the two above described applications are illustrated in Figure 5. The red line on the bottom of each section, shows that the algorithm detected all the real user taps (on the left part), while it remained flat on all simulated taps combined with vibration (on the right side). Thus, the movement caused by a human user tap can be accurately differentiated by our algorithm from the ones generated by vibration in all tested smartphones. This confirm that the user's tap cannot be simulated by malware using the vibration motors and thus it can be used to distinguish between human and malware.

It is important to notice that in this experiment, the acceleration data were measured by the smartphone built-in sensor because the goal of this section is only to show the difference between the measured data when the human tap his finger on the touch-screen and the smartphone vibration. Thus, which sensor has been used to measure these data is not important. However, this does not remove the requirement of using a trusted sensor to implement the Invisible CAPPCHA scheme in the real world.

As previously mentioned, Invisible CAPPCHA can be used by any web/cloud

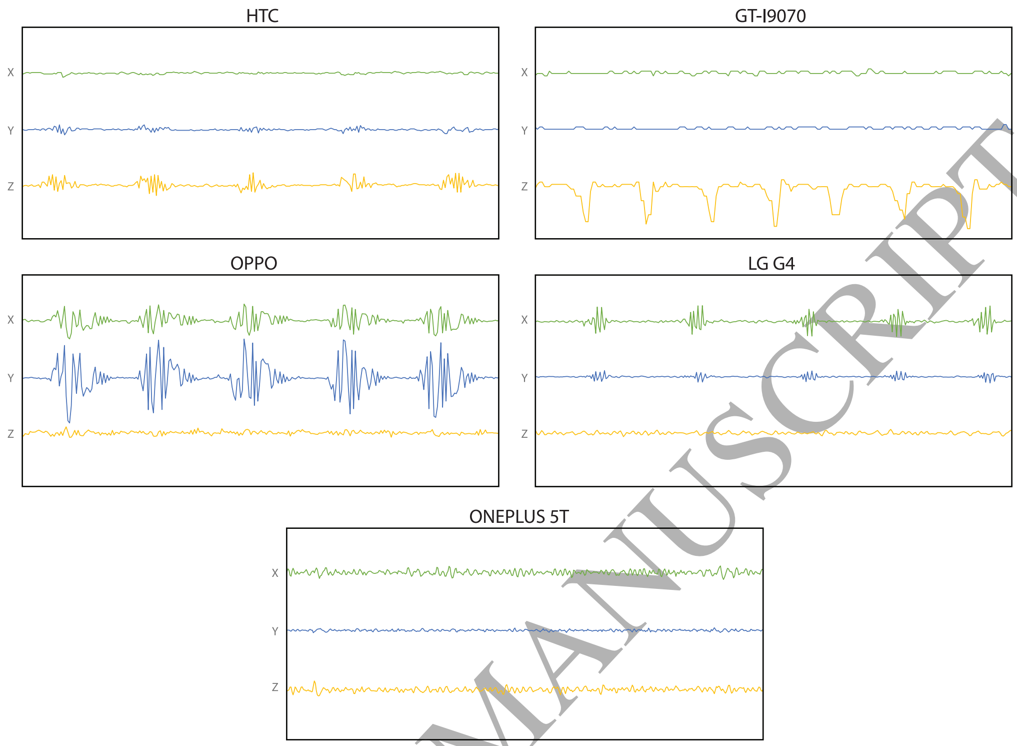


Figure 4: The effect of smartphone built-in vibration motor on acceleration change in GALAXY S ADVANCE, HTC DESIRE, OPPO F1, LG G4 and ONEPLUS 5T.

application that needs to be guaranteed that it's dealing with human user and input (e.g., a tap). In order to implement Invisible CAPPCHA, a secure element embedded with motion sensor such as SIMSense [38] is required to ensure the security of the sensory data, while the integrated tap detection algorithm analyzes these data to distinguish between humans and malware. As the SIMsense secure element is available as a SIM card, it can be used inside any mobile device without requiring any hardware change. Furthermore, the communication between the web/cloud applications and the secure element is carried out via the web API defined by Global Platform [48].

4. Threat model

We assume that Invisible CAPPCHA uses the web API defined by Global Platform [48] to access the secure element. Although the secure element may

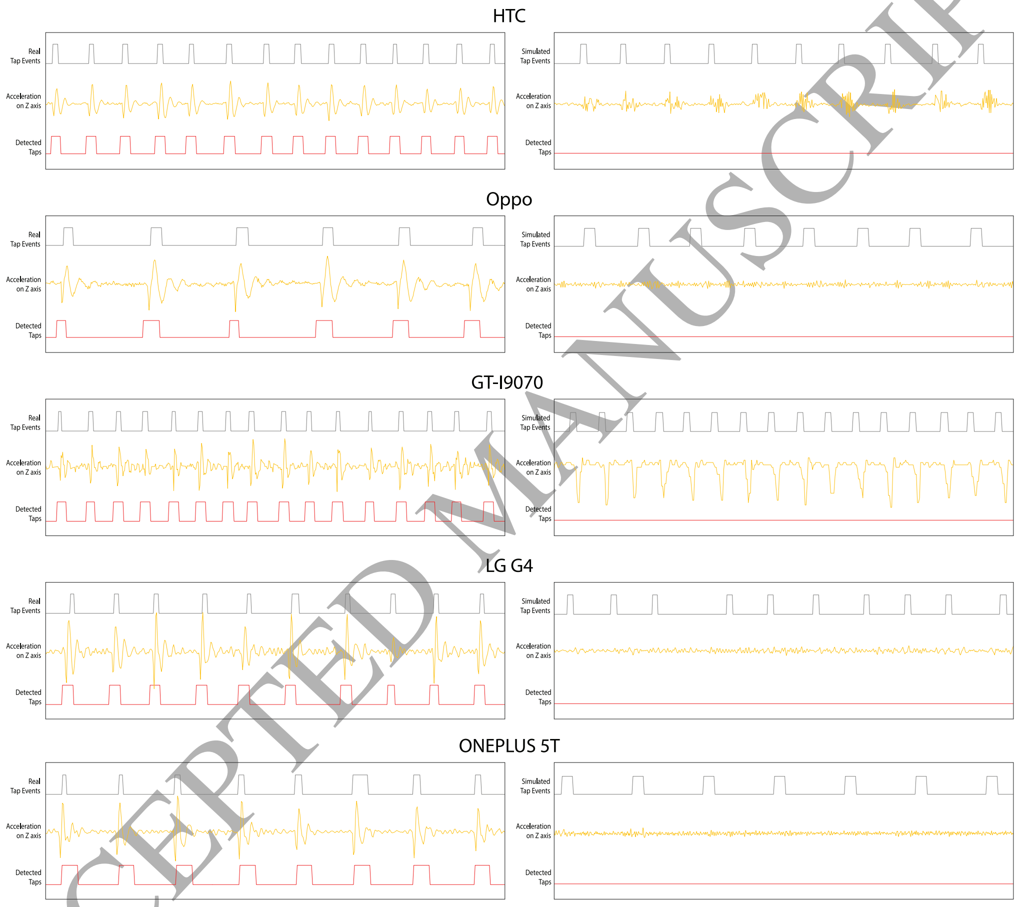


Figure 5: Comparison between tap and vibration acceleration patterns along the Z axis.

bring additional security to a cloud application/service, it is not, in itself, capable to fully ensure that the application is secure. In particular, we may identify threats in the following categories:

- Communication
- Access
- Sensitive Data Exposure

In the first category, several security threats may affect the proposed scheme, starting from a malicious entity generating and injecting forged verifications, or tampering with transmitted verification messages or finally, collecting successful legitimate verification and re-playing them associated to automatically generated transactions. In fact, the communication between the cloud application/service and the secure element is not automatically secured and thus, a security mechanism such as encryption has to be implemented to ensure the integrity and authenticity of the exchanged messages. In our security model, the secure element signs the result of Invisible CAPPCHA verification before sending it to the cloud application/service at the server-side. However, this by itself is not sufficient to prevent a replay attack.

In the second category, the access to the application stored in the secure element usually requires authentication of the off-card communicating party, for instance by asking user to present a PIN/password to unlock access. Thus, if malware has sufficient privileges to access smart card services, it can impersonate a legitimate application and try to brute-force this password. However, as in general only three attempts are allowed before the secure element is blocked, a brute force attack is not possible. Nonetheless, more than three invalid PIN/password values represent a denial of service attack as any further interaction with the secure element will be denied. The malware could also steal the user's password through side channel attack [37] and replay it to gain unauthorized access.

Finally, it is important to notice that for our threat model the trustworthiness of the server (the service provider) is irrelevant. Invisible CAPPCHA only provides the information about the fact that the interacting entity at the device side is human or not, no sensitive data are provided to the server side as the interpretation of the accelerometer data is completely performed inside the secure element.

5. Security Analysis

In this section, we discuss the security of our model against the aforementioned threats and other known attacks.

5.1. Replay attack (Communication to the Server)

In our security model, the secure element signs the result of an Invisible CAPPCHA verification before sending it to the cloud application/service (server).

The risk of replay attacks, where the malware can steal the signed message of valid input performed by the human and replay this message to gain unauthorized access to the protected service, has to be addressed by sending a signed unique value, i.e., a nonce, with the signed result of the Invisible CAPPCHA verification.

In detail, according to the well-known ECDSA scheme [49], after choosing an elliptic curve group of order q with a base point (generator) G laying on the curve, the pair (m, n) constituted by the verification result message m and a nonce n is hashed by using SHA-256 to a bitstring of length no more than the bit length of q, which is then transformed to an integer e:

$$e = leftmostbits(|q|, \mathrm{SHA256}(m||n)) \tag{2}$$

where "$||$" represents the bitwise concatenation operator and $|q|$ is the bit-length of the group order q. Then, a cryptographically secure pseudorandom integer k ranging from 1 to $q-1$ is selected to determine the point $(x_1, y_1) = k \cdot G$ on the curve, the x_1 is transformed to an integer to compute $r = x_1 \ mod \ q$. The signature of the pair (m, n) is the pair (r, s) of integers modulo q, where $s = k^{-1}(e + d \cdot r) \ mod \ q$ where d is the private key used for signing. Another elliptic curve point $Q = d \cdot G$ will be the public key available to the server for verification, performed through the same secure hash algorithm as in the signature process, so that the message digest signed by the authenticator is computed which, together with the public key Q and the digital signature components r and s, leads to the result. Finally, the signed message sent to the server will be composed by the 4-tuple (m, n, r, s) where the last two elements will be used for signature verification.

5.2. Reverse engineering attack

We assume that the JavaScript code of cloud application/service running in the browser is obfuscated to transform the code into a new representation that is harder to understand, copy, re-use and modify without authorization. However, successful de-obfuscation could be achieved in practice. In our security model, even if an attacker could de-obfuscate the code and remove or change the code related to the communication with the secure element to validate Invisible CAPPCHA, it will not succeed in the attack as the server allows access to the protected service only if the message that indicates that the input has been performed by a human is digitally signed by the secure element. Thus, the attacker cannot bypass Invisible CAPPCHA using reverse engineering.

5.3. Human-solver relay attacks

Human-solver relay attacks consist of relaying CAPTCHA challenges to remote human-solvers to bypass the security provided by CAPTCHAs. As CAPTCHA aims to distinguish between human and malware, it does not make difference between the legitimate user and remote human-solver. This is why this attack remains the most effective against the most, if not all, existing CAPTCHAs. However, Invisible CAPPCHA is transparent and does not require any additional task. Therefore, there is not a challenge to send to the human solvers. Thus, we argue that Invisible CAPPCHA ensures the security against human solver attack compared with the traditional CAPTCHAs.

5.4. Brute force and password replay attacks (Access to the Secure Element)

A brute force attack is an attempt to discover a password by trying all possible combination of letters, numbers and symbols until the password is found. In order to prevent malicious application from discovering the password through brute force, we suggest to use Invisible CAPPCHA to validate any input before it is used as a password candidate. In the case of a valid password typed by the human, the secure element provides access. Otherwise, the number of remaining attempts is decremented. In this way, even if a malware manages to guess the right password in the first attempt or steals the password through a side-channel attack, e.g., [33, 36, 37, 50] , it will not be able to gain access to the secure element as Invisible CAPPCHA will stop it. This way, Invisible CAPPCHA enhances the security of password against brute force and replay attacks.

5.5. Denial of service attack (Access to the Secure Element)

Usually a Secure Element allows three attempts before blocking the access. Thus, a malicious code can present multiple invalid passwords to block access to the Secure Element and enact a Denial of Service attack. Invisible CAPPCHA can be used to mitigate this attack as it can prevent malware from submitting passwords to the secure element. Therefore, the Secure Element can block the access to itself only if three invalid passwords have been entered by an Invisible CAPPCHA certified human, while invalid password entered by a malicious application will simply be blocked before being used.

6. Conclusions

CAPTCHAs are mainly used to prevent fraudsters from conducting automated actions at a large scale and abuse of mobile cloud services; however, the discerning factor adopted (i.e., a cognitive task) makes CAPTCHAs very often inconvenient and hard to solve even for human subjects. In this paper, we proposed Invisible CAPPCHA, a new transparent version of CAPPCHA that leverages the physical nature of humans instead of hard cognitive tasks. Unlike the original CAPPCHA, the novel version presented in this paper does not require any additional step or task; in fact, it uses the micro-movements generated naturally during the user's input in webpages to prove that the user is human. The experimental results on five smartphones from different generations and equipped with diverse hardware show that the acceleration changes during a tap event follows certain pattern that programs cannot mimic through vibration. This way, we demonstrated that the micro-movement caused by user's tap captured by a trusted sensor can be used to tell computers and humans apart. It is important to notice that when the smartphone is sitting on a hard surface such as a table, Invisible CAPPCHA has a low level of accuracy in the detection of the tap event. For this reason, when the test starts, Invisible CAPPCHA checks if the smartphone is in the user's hand. Furthermore, as the main purpose of Invisible CAPPCHA is to prove that the current action is requested by a human user, to impose that the user holds the smartphone in his hand does not represent a strong limitation; in fact, users usually leave their smartphone on tables or other surfaces only when they do not need it. In future work, we plan to develop an advanced tap detection algorithm capable of being effective in all common conditions of smartphone's use and test it with a higher number of participants. In addition, we plan to test the effectiveness of Invisible CAPPCHA with different form factors such as tablets and smart watches. Finally, the feasibility to leverage Invisible CAPPCHA failures to detect the presence of malware on the device and notify the user of such a problem will be studied.

References

[1] E. Ferrara, O. Varol, C. Davis, F. Menczer, A. Flammini, The rise of social bots, Commun. ACM 59 (7) (2016) 96–104. doi:10.1145/2818717.
URL `http://doi.acm.org/10.1145/2818717`

[2] C. Xiang, F. Binxing, Y. Lihua, L. Xiaoyi, Z. Tianning, Andbot: Towards advanced mobile botnets, in: Proceedings of the 4th USENIX Conference on Large-scale Exploits and Emergent Threats, LEET'11, USENIX Association, Berkeley, CA, USA, 2011, pp. 11–11.
URL `http://dl.acm.org/citation.cfm?id=1972441.1972456`

[3] M. Eslahi, R. Salleh, N. B. Anuar, Mobots: A new generation of botnets on mobile devices and networks, in: 2012 International Symposium on Computer Applications and Industrial Electronics (ISCAIE), 2012, pp. 262–266. doi:10.1109/ISCAIE.2012.6482109.

[4] D. Chasaki, C. Mansour, Int. J. of Space-Based and Situated Computing 5 (3) (2015) 141–149.

[5] L. von Ahn, B. Maurer, C. McMillen, D. Abraham, M. Blum, recaptcha: Human-based character recognition via web security measures, Science 321 (5895) (2008) 1465–1468. arXiv:http://science.sciencemag.org/content/321/5895/1465.full.pdf, doi:10.1126/science.1160379.
URL `http://science.sciencemag.org/content/321/5895/1465`

[6] I. J. Goodfellow, Y. Bulatov, J. Ibarz, S. Arnoud, V. D. Shet, Multi-digit number recognition from street view imagery using deep convolutional neural networks, CoRR abs/1312.6082. arXiv:1312.6082.
URL `http://arxiv.org/abs/1312.6082`

[7] C. Cruz-Perez, O. Starostenko, F. Uceda-Ponga, V. Alarcon-Aquino, L. Reyes-Cabrera, Breaking recaptchas with unpredictable collapse: Heuristic character segmentation and recognition, in: J. A. Carrasco-Ochoa, J. F. Martínez-Trinidad, J. A. Olvera López, K. L. Boyer (Eds.), Pattern Recognition, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 155–165.

[8] O. Starostenko, C. Cruz-Perez, F. Uceda-Ponga, V. Alarcon-Aquino, Breaking text-based captchas with variable word and character orientation, Pattern Recognition 48 (4) (2015) 1101 – 1112. doi:https://doi.org/10.1016/j.patcog.2014.09.006.
URL `http://www.sciencedirect.com/science/article/pii/S0031320314003483`

[9] Google Inc., recaptcha: Easy on humans, hard on bots (January 2018).
URL `https://www.google.com/recaptcha/intro/`

[10] Google Inc., Introducing the invisible recaptcha! (January 2018).
URL `https://www.google.com/recaptcha/intro/invisible.html`

[11] M. Guerar, M. Migliardi, A. Merlo, M. Benmohammed, B. Messabih, A completely automatic public physical test to tell computers and humans apart: A way to enhance authentication schemes in mobile devices, in: 2015 International Conference on High Performance Computing Simulation (HPCS), 2015, pp. 203–210. doi:10.1109/HPCSim.2015.7237041.

[12] M. Guerar, A. Merlo, M. Migliardi, Completely automated public physical test to tell computers and humans apart: A usability study on mobile devices, Future Generation Computer Systems-doi:https://doi.org/10.1016/j.future.2017.03.012.
URL `http://www.sciencedirect.com/science/article/pii/S0167739X17303709`

[13] S. Sivakorn, J. Polakis, A. D. Keromytis, I'm not a human: Breaking the google recaptcha, 2016.

[14] N. M. Al-Fannah, Using aesthetic judgements to distinguish between humans and computers, CoRR abs/1704.02972.

[15] B. Pinkas, T. Sander, Securing passwords against dictionary attacks, in: Proceedings of the 9th ACM Conference on Computer and Communications Security, CCS '02, ACM, New York, NY, USA, 2002, pp. 161–170. doi:10.1145/586110.586133.
URL `http://doi.acm.org/10.1145/586110.586133`

[16] H. Gao, X. Liu, A new graphical password scheme against spyware by using captcha, in: Proceedings of the 5th Symposium on Usable Privacy

and Security, SOUPS '09, ACM, New York, NY, USA, 2009, pp. 21:1–21:1. doi:10.1145/1572532.1572560.
URL `http://doi.acm.org/10.1145/1572532.1572560`

[17] L. Wang, X. Chang, Z. Ren, H. Gao, X. Liu, U. Aickelin, Against spyware using captcha in graphical password scheme, in: 2010 24th IEEE International Conference on Advanced Information Networking and Applications, 2010, pp. 760–767. doi:10.1109/AINA.2010.46.

[18] H.-T. Yeh, B.-C. Chen, Y.-C. Wu, Mobile user authentication system in cloud environment, Security and Communication Networks 6 (9) (2013) 1161–1168. doi:10.1002/sec.688.
URL `http://dx.doi.org/10.1002/sec.688`

[19] D. Pequegnot, L. Cart-Lamy, A. Thomas, T. Tigeon, J. Iguchi-Cartigny, J. L. Lanet, A security mechanism to increase confidence in m-transactions, in: 2011 6th International Conference on Risks and Security of Internet and Systems (CRiSIS), 2011, pp. 1–8. doi:10.1109/CRiSIS.2011.6061836.

[20] I. A. Althamary, E. S. M. El-Alfy, A more secure scheme for captcha-based authentication in cloud environment, in: 2017 8th International Conference on Information Technology (ICIT), 2017, pp. 405–411. doi:10.1109/ICITECH.2017.8080034.

[21] G. Moy, N. Jones, C. Harkless, R. Potter, Distortion estimation techniques in solving visual captchas, in: Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004., Vol. 2, 2004, pp. II–23–II–28 Vol.2. doi:10.1109/CVPR.2004.1315140.

[22] A. A. Chandavale, A. M. Sapkal, R. M. Jalnekar, Algorithm to break visual captcha, in: 2009 Second International Conference on Emerging Trends in Engineering Technology, 2009, pp. 258–262. doi:10.1109/ICETET.2009.24.

[23] J. Yan, A. S. El Ahmad, A low-cost attack on a microsoft captcha, in: Proceedings of the 15th ACM Conference on Computer and Communications Security, CCS '08, ACM, New York, NY, USA, 2008, pp.

543–554. doi:10.1145/1455770.1455839.
URL http://doi.acm.org/10.1145/1455770.1455839

[24] M. Korakakis, E. Magkos, P. Mylonas, Automated captcha solving: An empirical comparison of selected techniques, in: 2014 9th International Workshop on Semantic and Social Media Adaptation and Personalization, 2014, pp. 44–47. doi:10.1109/SMAP.2014.29.

[25] G. Reynaga, S. Chiasson, The usability of captchas on smartphones, in: 2013 International Conference on Security and Cryptography (SECRYPT), 2013, pp. 1–8.

[26] A. Merlo, M. Migliardi, P. Fontanelli, On energy-based profiling of malware in android, 2014, pp. 535–542. doi:10.1109/HPCSim.2014.6903732.

[27] B. Shrestha, N. Saxena, J. Harrison, Wave-to-access: Protecting sensitive mobile device services via a hand waving gesture, in: M. Abdalla, C. Nita-Rotaru, R. Dahab (Eds.), Cryptology and Network Security, Springer International Publishing, 2013, pp. 199–217.

[28] M. Guerar, M. Migliardi, A. Merlo, M. Benmohammed, F. Palmieri, A. Castiglione, Using screen brightness to improve security in mobile social network access, IEEE Transactions on Dependable and Secure Computing PP (99) (2016) 1–1. doi:10.1109/TDSC.2016.2601603.

[29] M. Conti, I. Zachia-Zlatea, B. 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, ACM, New York, NY, USA, 2011, pp. 249–259. doi:10.1145/1966913.1966945.
URL http://doi.acm.org/10.1145/1966913.1966945

[30] A. Buriro, S. Gupta, B. Crispo, Evaluation of motion-based touch-typing biometrics for online banking, in: 2017 International Conference of the Biometrics Special Interest Group (BIOSIG), 2017, pp. 1–5. doi:10.23919/BIOSIG.2017.8053504.

[31] A. De Luca, A. Hang, F. Brudy, C. Lindner, H. Hussmann, Touch me once and i know it's you!: Implicit authentication based on touch screen patterns, in: Proceedings of the SIGCHI Conference on Human Factors

in Computing Systems, CHI '12, ACM, New York, NY, USA, 2012, pp. 987–996. doi:10.1145/2207676.2208544.
URL `http://doi.acm.org/10.1145/2207676.2208544`

[32] L. Cai, H. Chen, Touchlogger: Inferring keystrokes on touch screen from smartphone motion, in: Proceedings of the 6th USENIX Conference on Hot Topics in Security, HotSec'11, USENIX Association, Berkeley, CA, USA, 2011, pp. 9–9.
URL `http://dl.acm.org/citation.cfm?id=2028040.2028049`

[33] Z. Xu, K. Bai, S. Zhu, Taplogger: Inferring user inputs on smartphone touchscreens using on-board motion sensors, in: Proceedings of the Fifth ACM Conference on Security and Privacy in Wireless and Mobile Networks, WISEC '12, ACM, New York, NY, USA, 2012, pp. 113–124. doi:10.1145/2185448.2185465.
URL `http://doi.acm.org/10.1145/2185448.2185465`

[34] E. Miluzzo, A. Varshavsky, S. Balakrishnan, R. R. Choudhury, Tapprints: Your finger taps have fingerprints, in: Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services, MobiSys '12, ACM, New York, NY, USA, 2012, pp. 323–336. doi:10.1145/2307636.2307666.
URL `http://doi.acm.org/10.1145/2307636.2307666`

[35] E. Owusu, J. Han, S. Das, A. Perrig, J. Zhang, Accessory: Password inference using accelerometers on smartphones, in: Proceedings of the Twelfth Workshop on Mobile Computing Systems &#38; Applications, HotMobile '12, ACM, New York, NY, USA, 2012, pp. 9:1–9:6. doi:10.1145/2162081.2162095.
URL `http://doi.acm.org/10.1145/2162081.2162095`

[36] A. J. Aviv, B. Sapp, M. Blaze, J. M. Smith, Practicality of accelerometer side channels on smartphones, in: Proceedings of the 28th Annual Computer Security Applications Conference, ACSAC '12, ACM, New York, NY, USA, 2012, pp. 41–50. doi:10.1145/2420950.2420957.
URL `http://doi.acm.org/10.1145/2420950.2420957`

[37] M. Mehrnezhad, E. Toreini, S. F. Shahandashti, F. Hao, Touchsignatures: Identification of user touch actions and pins based on mobile

sensor data via javascript, Journal of Information Security and Applications 26 (2016) 23 – 38. doi:https://doi.org/10.1016/j.jisa.2015.11.007.
URL `http://www.sciencedirect.com/science/article/pii/S2214212615000678`

[38] Oberthur Technologies, Oberthur technologies teams up with vivo to provide simsense to mobile subscribers in brazil (February 2010).
URL `https://www.edn.com/electronics-products/electronic-product-releases/other/4202585/Oberthur-Technologies-Teams-up-With-Vivo-to-Provide-SIMSense-to-Mobile-Subscr`

[39] U.S. Department of Commerce/National Institute of Standards and Technology, Digital signature standard (dss). fips-186-4. (2013).
URL `http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf`

[40] N. Koblitz, Elliptic curve cryptosystems, Mathematics of computation 48 (177) (1987) 203–209.

[41] V. S. Miller, Use of elliptic curves in cryptography, in: Conference on the theory and application of cryptographic techniques, Springer, 1985, pp. 417–426.

[42] Global Platform's White Paper, The trusted execution environment: Delivering enhanced security at a lower cost to the mobile market, Tech. rep. (2011).

[43] Apple, Uiacceleration class reference. (2016).
URL `https://developer.apple.com/library/ios/documentation/uikit/reference/UIAcceleration_Class/Reference/UIAcceleration.html`

[44] S. Heo, G. Lee, Forcetap: Extending the input vocabulary of mobile touch screens by adding tap gestures, in: Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services, MobileHCI '11, ACM, New York, NY, USA, 2011, pp. 113–122. doi:10.1145/2037373.2037393.
URL `http://doi.acm.org/10.1145/2037373.2037393`

[45] E. Davarci, B. Soysal, I. Erguler, S. O. Aydin, O. Dincer, E. Anarim, Age group detection using smartphone motion sensors, in: 2017 25th

European Signal Processing Conference (EUSIPCO), 2017, pp. 2201–2205. doi:10.23919/EUSIPCO.2017.8081600.

[46] K. Tuck, Mma8450q single/double and directional tap detection, Tech. rep. (2010).
URL https://www.nxp.com/docs/en/application-note/AN3919.pdf

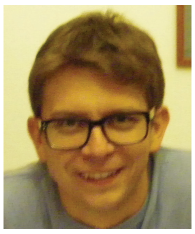
[47] Google Inc., Android api: Sensor class. (2018).
URL https://developer.android.com/reference/android/hardware/Sensor.html

[48] GlobalPlatform, Web api for accessing secure element. (2016).
URL https://globalplatform.github.io/WebApis-for-SE/doc/

[49] D. Johnson, A. Menezes, S. Vanstone, The elliptic curve digital signature algorithm (ecdsa), International journal of information security 1 (1) (2001) 36–63.

[50] L. Simon, R. Anderson, Pin skimmer: Inferring pins through the camera and microphone, in: Proceedings of the Third ACM Workshop on Security and Privacy in Smartphones &#38; Mobile Devices, SPSM '13, ACM, New York, NY, USA, 2013, pp. 67–78. doi:10.1145/2516760.2516770.
URL http://doi.acm.org/10.1145/2516760.2516770

Meriem Guerar received the Master degree and PhD from the University of Sciences and the Technology of Oran (USTO), Algeria. She also worked as a post-doctoral researcher at the University of Padova and is now with the University of Genova. Her main research interests include the areas of authentication and identity management, security and usability, smartphone security and payment systems security.



Alessio Merlo got a MSc. in Computer Science in 2005 at University of Genova. He received his PhD in Computer Science from University of Genova (Italy) in 2010 where he worked on performance and access control issues related to Grid Computing. He is currently serving as an Assistant Professor at the Universita degli Studi di Genova, Italy, where he collaborates with the CSec Lab at DIBRIS. His currently research interests are focused on performance and security issues related to Web, distributed systems (Grid, Cloud) and mobile (Android platform). He is a member of the IEEE Computer Society and ACM. He participates to program committees of international conferences and is member of the Editorial Board of an International Journal (Journal of High Speed Networks).

Mauro Migliardi got his PhD in Computer Engineering in 1995. He was a

Research Associate and Assistant Professor at the University of Genoa and Research Associate at Emory University; currently he is Associate Professor at the University of Padua, Adjunct Professor at the University of Genoa, member of the Steering Committee of the Center for Computing Platforms Engineering and of the Scientific Board of Circle Garage s.r.l. start-up. His main research interest is distributed systems engineering, with a focus on security, pervasive systems, human memory support services and energy awareness. He has won the 2013 Canada-Italy Innovation Award, tutored more than 100 among Bachelor, Master and PhD students at the Universities of Genoa, Padua and Emory, and (co-)authored more than 120 scientific papers.

Francesco Palmieri received the M.S. degree and the Ph.D. degree in computer science from the University of Salerno. Currently, he is an Associate Professor at the University of Salerno. Previously, he has been an Assistant Professor at the Second University of Naples, and the Director of the telecommunication and networking division of the Federico II University, Naples. He has been closely involved in the development of the Internet in Italy as a Senior Member of the Technical-Scientific Advisory Committee and of the CERT of the Italian NREN GARR. His research interests include advanced networking protocols and architectures as well as network security. He has published a significant number of papers (more than 150) in leading technical journals, books, and conferences, and currently serves as the Editor-in-Chief of an international journal and is part of the Editorial Board (Associate Editor) of several other indexed ones. He also achieved several formal appreciations and service awards for the organization of international conferences and scientific events, where he covered several key roles.