ascCAPTCHA: an Invisible Sensor CAPTCHA for PCs Based on Acoustic Side Channel

R. Di Nardo Di Maio*, M. Guerar** and M. Migliardi*
* Universita' degli Studi di Padova, Padova, Italia
** Universita' degli Studi di Genova, Genova, Italia
mauro.migliardi@unipd.it

Abstract – **Our growing reliance on the digital world has caused a similar growth in the sophistication of bots trying to impersonate humans. The most classic tool to tell human and computers apart is the CAPTCHA, however CAPTCHAs based on cognitive challenges are becoming either insecure or very difficult to be solved by humans too. A possible solution is leveraging the rich sensor set of modern mobile devices to capture the physical nature of humans while they are interacting with the system, however, traditional PCs do not have the same opportunity. In this paper we describe ascCAPTCHA, a CAPTCHA based on an acoustic side-channel that leveraging a simple microphone is compatible with PCs lacking the rich sensor set of smart devices.**

Keywords - CAPTCHA, side-channel, acoustic, bot, sensor CAPTCHA, PC.

I. INTRODUCTION

Recently services delivered over the Internet have become part of the daily routine of everyone and a lot of private user's information is continuously uploaded on the network. This rich trove of data has attracted a lot of attention, and the development of malware has also grown and is now often empowered with advanced Artificial Intelligence techniques. A recent study [1] pointed out that about a fifth of all web traffic (i.e., 20.4%) comes from bad bots, which continuously attack websites, mobile apps, and services' APIs. These malicious bots perform activities such as registering thousands of free accounts, testing stolen usernames and passwords or credit card details at a large scale, obtaining tickets for resale, increasing click rates on pay-per-click ads, or stealing data from websites usually related to pricing, etc.

The most common mechanism used to prevent these malicious actions is called a Completely Automated Public Turing test to tell Computers and Humans Apart (CAPTCHA). However, sophisticated bots are able to mimic human behavior and bypass the most commonly adopted CAPTCHA mechanisms (e.g., no CAPTCHA reCAPTCHA, GEETest, etc) [2, 3, 4].

Recently, a new type of CAPTCHA based on sensor data has emerged to stop these bots. Unlike the traditional CAPTCHAs that leverage the cognitive capability and/or behavior of humans to identify them, these CAPTCHAs instead leverage their physical nature. The idea behind these emerging CAPTCHAs is that humans can perform simple physical tasks easily, while bots, being pieces of software, cannot. Classical examples of CAPTCHAs in this category are CAPPCHA [5,6] and Invisible CAPPCHA [7]. However, to the best of our knowledge, both of them, as well as all sensor-based CAPTCHAs proposed in the literature, are designed for mobile devices and can not be used in PCs due to the lack of a rich sensor set. This motivated us to design a new sensor-based CAPTCHA that leverages the microphone, a sensor that is nowadays available in almost every device, to tell humans and computers apart.

The rest of this paper is organized as follows. In section II, we provide an overview of the related work; in Section III we describe the ascCAPTCHA scheme; in Section IV we provide an overview of the ascCAPTCHA system, highlighting the ascCAPTCHA evaluation process and the messages exchanged between the application and the remote web service; in Section V we describe our experimental results; in Section VI we provide a security analysis of the proposed scheme against some well-known types of attacks; finally, in Section VII we draw our conclusions and point out potential directions for future work.

II. RELATED WORK

Recently, several sensor-based CAPTCHA mechanisms have been proposed for mobile devices, either to improve the usability of the traditional CAPTCHAs or to make them more resilient against the new bot generations. These mechanisms use the sensors either as an input mechanism to solve a cognitive task (e.g., AccCAPTCHA [8], GISCHA [9], SenCAPTCHA [10]) or to detect a physical task performed by the user [5,6, 7].

The first sensor-based CAPTCHA has been proposed by Liao et al.[8] in 2013. Their proposed method, named AccCAPTCHA, asks the users to play a simple rolling ball game by moving the device. They used the accelerometer as an input mechanism to move the ball. The users are considered as human if they successfully move the ball to the destination hole. In 2015, a new type of sensor-based CAPTCHA called CAPPCHA (a Completely Automatic Public Physical test to tell Computers and Humans Apart) [5,6] emerged which leverages the physical nature of the human to distinguish between the human and bots. To solve the CAPPCHA Challenge, users are required to tilt the smartphone to a specific degree. The motion sensors have been used to detect the physical task performed by the user. The same principle behind CAPPCHA has also been leveraged by Guerar el al. [11,12] to prevent automated programs from performing a brute force [13] or a credential stuffing attack [14]. The authors proposed two

authentication methods (i.e., 2GesturePIN [11, 15] and CirclePIN [12]) for smartwatches that embed a form of CAPPCHA. To log in, users have to physically rotate the bezel or the smartwatch digital crown to a specific degree to input the PIN digits. Similar to CAPPCHA, Hupperich et al. [16] proposed Sensor CAPTCHA which asks the users to perform a physical task to prove they are human such as hammering, fishing, drinking, or turning the body while holding the mobile device. Mantri et al. [17] proposed a CAPTCHA mechanism that asks the users to move the device according to a specific pattern displayed on the screen. For instance, the user is required to write an "M" letter while holding the device.

Invisible CAPPCHA [7] is the work that is most closely related to ours. The authors leverage the motion sensor readings to distinguish between humans and bots in a fully transparent way. When the users interact with the device to fill a form or write a comment, Invisible CAPPCHA analyses sensor data to detect the micro-movement of the device caused by the user's tap on the touchscreen. Since simulated tap events cannot generate the same movement pattern, this method can effectively separate humans from bots. The security and usability offered by Invisible CAPPCHA attracted a company called Brave, which developed the first prototype of a sensor-based CAPTCHA based on the same key concept of Invisible CAPPCHA [18]. Further details on conventional and emerging CAPTCHA schemes can be found in [23].

III. ASCCAPTCHA

Despite the variety of sensor-based CAPTCHA methods proposed in the literature, no one among them is compatible with PCs. This is mainly due to the fact that they are leveraging motion sensors such as the accelerometer and gyroscope, which are available only in smartphones, tablets, and smartwatches. This motivated us to propose a new sensor-based CAPTCHA compatible with PCs, yet usable in other devices as well. Our goal is to develop a secure and usable CAPTCHA mechanism compatible with devices of different form-factor.

Inspired by Invisible CAPPCHA [7] and acoustic side channel attacks [19, 20, 21], we propose ascCAPTCHA, a novel CAPTCHA mechanism based on Acoustic Side Channel. Similar to Invisible CAPPCHA, the ascCAPTCHA is fully transparent to the users and it leverages the side channel to detect the user's tap. The main difference is that the sensor data analysed by ascCAPTCHA is generated by the microphone, instead of the accelerometer.

In the past, acoustic side channel has been used to infer user's keystrokes typed on the touchscreen, keyboards, and ATM pad [19, 20], but this is the first time it is used in a CAPTCHA scheme to improve the security rather than compromise it.

In order to tell humans and computers apart, ascCAPTCHA analyses the sensor data captured by the microphone during interaction with online services that require protection against automation abuses (e.g., fill a form, write a comment, sign up, etc.). The rationale behind this is that the user's tap on the keyboard or screen is a physical interaction that generates a sound wave that can be detected easily by the microphone [19]. Fig. 1 illustrates the audio peaks detected when a human user types on a keyboard a word seven characters long. On the contrary, the simulated taps by the bot do not generate any sound. Unlike the traditional CAPTCHA methods, using ascCAPTCHA allows users to interact with the online services without the need to solve any challenge which makes it fully transparent and thus very usable.

IV. SYSTEM OVERVIEW

Similar to CAPTCHAs, ascCAPTCHA can be used to prevent any abuses of online services that require user's input. In this paper, we take authentication to a web-based service as a use case.

A. ascCAPTCHA evaluation process

During the evaluation of the user's input, ascCAPTCHA records two audio signals using the microphone: the first one just before the insertion of the password to define the noise threshold and a second one during the user's input. The noise threshold will be used to find significant audio sequences (i.e., audio peaks) in the second signal. We tested the following three approaches to differentiate between the humans and bots:

1. Time Correspondence

In this approach, ascCAPTCHA stores the timestamps of each character typed by the user, while recording the audio signal from the microphone at the same time. If at least 90% of the timestamps of the detected audio peaks overlap with the stored timestamps, then the user is considered as human; otherwise, the input is considered as a malicious one injected by a bot.

2. Character correspondence

Similar to acoustic side channel attacks[19, 20], in this approach, ascCAPTCHA uses the microphone as a side channel to detect the characters typed by the user. If the detected sequence of characters matches the password characters, the user is considered as human, otherwise a bot.

Similarly to Asonov and Agrawal [19], we leveraged deep learning to classify each key pressed. When a key is pressed by a user on the keyboard, it produces a variation of the audio signal, called press peak, for a time window of about 8-10 ms [19]. The signal shape in each window can be divided into the following three consecutive areas: i) ***Touch peak*** which is a peak in a window of 2-3 ms, caused by the user's finger when touching the key, ii) ***Noisy meaningless area***, and iii) ***Hit peak*** which is a peak in a window of 2-3 ms at the end of the 8-10 ms window, caused by the finger and the key hitting the keyboard supporting plate.

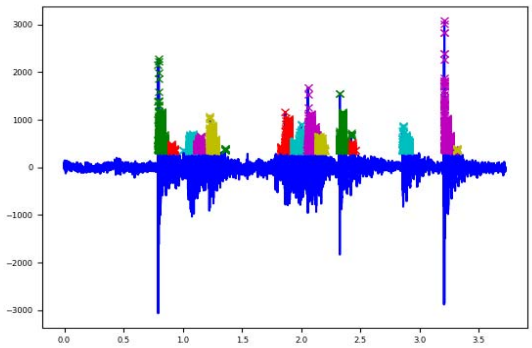

Figure 1. The audio peaks detected by ascCAPTCHA when a human user types a word seven characters long

- *Feature selection*

To predict a key from an audio peak, the types of features that can be used as input of the neural network are:

- Normalized FFT of only the touch peak (see Fig. 2)
- Normalized FFT of the touch peak concatenated with Normalized FFT of the hit peak (see Fig. 2)
- Output of VGG16 convolutional neural network, pre-trained on the ImageNet dataset and without the last fully connected layers, and applied on the spectrogram of the concatenation of the hit and the press peak (see Fig. 3)

- *Data collection and neural network training*

During the data collection phase, we recorded the audio signal of 200 clicks per key, then a time-shift was applied to them to perform data augmentation. From each peak in the recorded audio files, we extracted the feature with one of the previously mentioned methods. Then we created a neural network from scratch, composed of a number of input neurons equal to the size of the features used and a number of output neurons equal to the number of keys of the keyboard. In the network there is only one hidden layer of 100 or 1024 neurons, depending on the type of feature used. Each output neuron is related to a key of the keyboard and the possible value is a decimal number in the [0, 1] range. We trained the neural network using the features extracted from the recorded audio files. The predicted key is the key with the highest value in the output neuron.

- *Classification of Humanness*

This process consists of the detection of the audio peaks during the insertion of the password, the extraction of the features from every audio peak, and its classification through our trained neural network model.

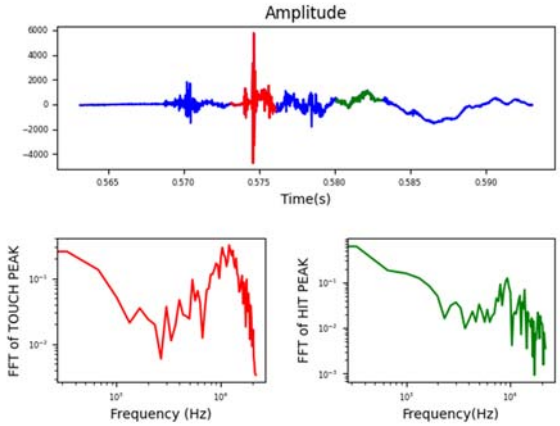

Figure 2. Example of normalized FFT of the hit peak and touch peak of a human tap of key '0'

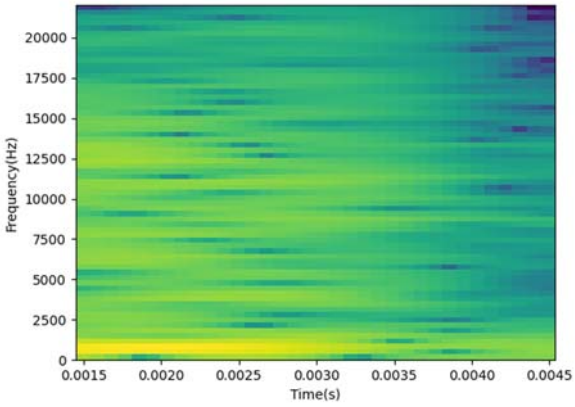

Figure 3. Example of the spectrogram computed for a human tap of key '0'

During the classification of each audio peak, we take the keys related to the 10 output neurons with the highest values instead of considering only the key with the best score. This is due to the fact that the correct key might be one of these 10 keys.

If each character of the password matches one of the characters in its corresponding set of predicted keys, the user is considered as human, otherwise a bot.

3. ***Time and character correspondence***

The character correspondence can be applied after the application of the time correspondence if the latter has positive results. The combination of the two approaches is more accurate and stronger against false positives (i.e., bot identified as human) than applying only the character correspondence but it is not better than the time correspondence approach by itself. Hence we do not use it.

B. Communication between client and server

ascCAPTCHA runs on the client side. The response of the evaluation will be signed through Elliptic Curve

Digital Signature Algorithm (ECDSA) and the credentials of the client will be sent to the server if and only if the user has been identified as human. The communication between the client and the server uses Transport Layer Security (TLS) protocol and follows these steps:

- ascCAPTCHA sends the message *(m, n, sign(m||n))* after the evaluation of the user's input, where: *m* is the result of the evaluation (*True* if a human user is detected, *False* otherwise); *n* is a nonce, a unique and random generated sequence which is very important to prevent replay attacks; *sign(m||n)* is the ECDSA signature of the concatenation of *m* with *n*.
- The server checks the integrity of the message. If the message has not been altered and the value of *m* is *True*, the server sends an "ok" message.
- ascCAPTCHA sends the user's credentials when the "ok" message is received. If the credentials are correct, the server allows the user to access the web service. The user has a limited number of attempts. If the maximum amount of trials has been reached, the ascCAPTCHA will block the next access of the user to the authentication service..

The different steps of communication between the client and the server are illustrated in Fig. 4.

V. Experimental results

All the tests were performed on a MSI GL63 8RD laptop, using the 2004 version of Windows 10 Home Operating System. It is important to mention that this laptop is equipped with a silent keyboard and even with a soft click, this has not an impact on ascCAPTCHA performance. ascCAPTCHA results are described in the following sections.

C. Detection of human users

We notice that using the character correspondence which is based on deep learning there are many false negatives given by the similar sound produced by several keys. This is why in this section, we present the results of time correspondence only. During the test, we set the maximum number of authentication attempts to 3. Then, we tested ascCAPTCHA 100 times for each precision percentage (i.e., 85%, 90%, 95%, 100%) using the following password: "*he35ghibn564st"*.

TABLE I. Accuracy of human detection with time correspondence

		Time correspondence percentage			
		100%	*95%*	*90%*	*85%*
Number of trial	*1st trial*	80 %	80%	83 %	80 %
	2nd trial	19 %	19 %	17 %	20 %
	3rd trial	1 %	1 %	0 %	0 %

As shown in Table 1, ascCAPTCHA usually detects the human presence at the first or second trial. The reason the human user was not identified as a human from the first trial is due to the background noise in the environment.

D. Detection of bots

In this section we discuss only the results obtained using the time correspondence approach because the character correspondence approach generates many false positives.

ascCAPTCHA was tested using simulated user's input 10 times. The username and password used to simulate the user authentication to a web service are (username: RaffaDNDM, password:"he35ghibn564st"). In all the authentication attempts, the input was identified as malicious injected by a bot. The audio files recorded by ascCAPTCHA were very similar and highlight the two most probable situations:

- *The noise during the noise evaluation (i.e., recorded just before the authentication) is very high:* In this case, no audio peak can be detected during the authentication (see Fig. 5).
- *The noise during the noise evaluation is very low:* In this case, some audio peaks caused by the background noise during the authentication can be detected. However, the timestamps of these audio peaks don't match the password characters' timestamps (see Fig. 6).

  The only case, in which a bot can successfully authenticate, is when there is a sequence of audio peaks caused by the noise, and the time between them is the same between the password characters. This event is extremely unlikely for passwords of non-trivial length because it is hard that there would be a high noise only during the insertion of the password and not during the noise evaluation. If no noise evaluation was performed, the bot could also analyse the background noise of the user and possibly identify a periodic sequence of peaks. Then type the characters with the same time intervals discovered in the previous noise pattern. This highlights the strength and the importance of the noise evaluation performed by ascCAPTCHA before the insertion of the password.

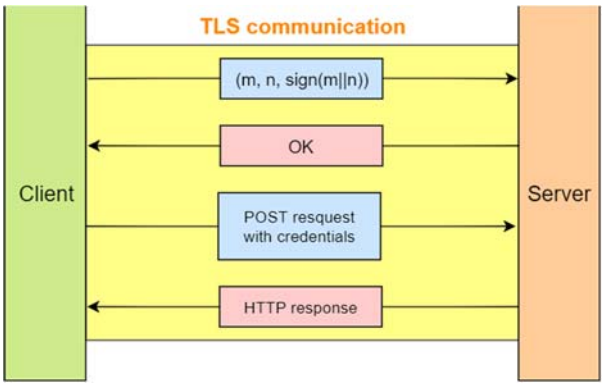

Figure 4. Communication between client and server

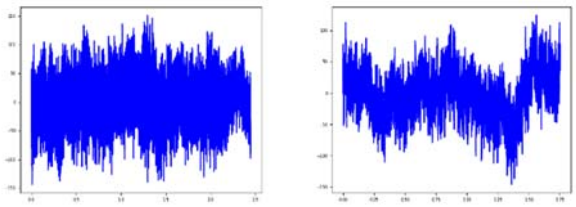

Figure 5. Audio peaks, detected during the password insertion of a bot, when the noise is very high during the noise evaluation

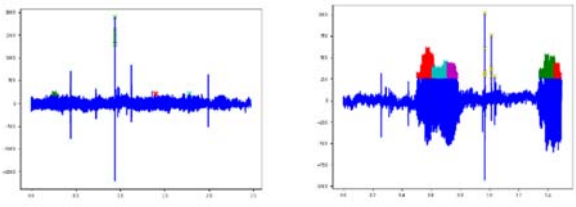

Figure 6. Audio peaks, detected during the password insertion of a bot, when the noise is very low during the noise evaluation

E. Other input devices

In order to test the effectiveness of the time correspondence approach, we tested ascCAPTCHA with other input devices (i.e., external wireless keyboard Logitech K480, mouse, and touchpad). In the first experiment, the external keyboard has been positioned in front of the laptop and the embedded microphone of the laptop has been used for recording the audio signal.

In the second experiment, the virtual keyboard of Windows 10 has been used with the mouse *(Zelotes T90)* and the touchpad as input devices. In the case of the mouse, the time correspondence leverages the sound of each mouse click used to select a key on the virtual keyboard. While, in the case of the touchpad, it leverages the sound of each click on the physical left button of the touchpad.

The results of the time correspondence using these devices were equal to the ones obtained using the laptop keyboard. Hence the time correspondence approach is very efficient and it works well in several scenarios. This is because it depends on the position of the microphone and not on the device used for the input. On the contrary, character correspondence is limited to the hardware characteristics of the input device.

VI. Security Analysis

In this section, we discuss the security of ascCAPTCHA against the following well-known attacks:

F. Replay attack

The message sent to the server which includes the ascCAPTCHA verification results along with a unique random number, called nonce, is signed using ECDSA. This is an effective way to prevent replay attacks because the server refuses any message with a previously used nonce. If the bot tries to change the nonce, the signature becomes invalid.

G. Reverse engineering attack

The code working on the client must be kept secure by the File System. Even if the attacker can reverse code on the server, the signature of the message and the TLS communication guarantee that this attack cannot be performed.

H. Human-solver relay attack

ascCAPTCHA is strong against this attack because there are no additional tasks to be performed as in traditional CAPTCHAs. Hence, no challenges can be sent to remote human solvers.

I. Brute force and credential stuffing attacks

Today, two of the biggest problems that threaten every website with a login are the use of malicious bots for credential stuffing and credential cracking using brute force attacks. In credential stuffing attacks, bots use the stolen credential sets (i.e., username and passwords) to authenticate at one or more services. While, in brute force attack, the bot tries all the possible character combinations until the password is found.

ascCAPTCHA is efficient against these attacks. In the case the user has been identified as a bot, ascCAPTCHA immediately blocks the user. While in the case that the user has been identified as a human, ascCAPTCHA provides the user three chances to input the correct credentials.

J. Denial Of Service (DOS)

If a bot is detected by ascCAPTCHA, ascCAPTCHA blocks it at the client side. Hence the server cannot be overloaded, because the client prevents bot generated attempts without communicating any message to the server.

K. Eavesdropping attack

In order to prevent this attack, all the messages exchanged between the client and the server, including the POST requests, are padded with space characters. This prevents an eavesdropper from understanding which string is sent in every step of the communication. In fact, if padding is not applied, the eavesdropper can analyse the size of the packets to understand which data are exchanged between the parties [22].

VII. Conclusion

Recently, a new generation of bad bots has emerged which represents a serious threat to all businesses in general. Sensor CAPTCHAs based on the physical nature of humans are one of the promising solutions to defend against these emerging bots. However, the proposals in the literature are not compatible with PCs. In this paper, we proposed ascCAPTCHA, a new invisible CAPTCHA that is compatible with all devices with a microphone, including PCs. In fact, our scheme leverages the sound generated naturally during the user's input to identify the human. The experimental results show that the time correspondence is the best method to evaluate the user's input. Its effectiveness has been tested using different input

devices. In future work, we plan to test ascCAPTCHA in other devices enabled with a microphone such as smartphones and smartwatches.

REFERENCES

[1] K. Gäwe, How To Protect Your Business Website from Bad Bots, 2020.

[2] Radware (2020), The big bad bot problem, 2020.

[3] B. Zhao, H. Weng, S. Ji, J. Chen, T. Wang, Q. He, and R. Beyah, Towards evaluating the security of real-world deployed image captchas, Proceedings of the 11th ACM Workshop on Artificial Intelligence and Security, ser. AISec '18. New York, NY, USA: Association for Computing Machinery, p. 85–96, 2018.

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

[5] M. Guerar, A. Merlo, and M. Migliardi, Completely automated public physical test to tell computers and humans apart: A usability study on mobile devices, Future Generation Computer Systems, vol. 82, pp. 617-630, 2018.

[6] M. Guerar, M. Migliardi, A. Merlo, M. Benmohammed, and B. Mess- abih, A completely automatic public physical test to tell computers and humans apart: A way to enhance authentication schemes in mobile devices, 2015 International Conference on High Performance Computing Simulation (HPCS), July 2015, pp. 203–210.

[7] M. Guerar, A. Merlo, M. Migliardi, and F. Palmieri, Invisible CAPPCHA: A usable mechanism to distinguish between malware and humans on the mobile IoT, Computers & Security, vol. 78, pp. 255-266, 2018.

[8] C.-J. Liao, C.-J. Yang, J.-T. Yang, H.-Y. Hsu, and J.-W. Liu, A game and accelerometer-based captcha scheme for mobile learning system, Proceedings of EdMedia + Innovate Learning 2013, J. Herrington, A. Couros, and V. Irvine, Eds. Victoria, Canada: Association for the Advancement of Computing in Education (AACE), June 2013, pp. 1385–1390.

[9] T.-I. Yang, C.-S. Koong, and C.-C. Tseng, Game-based image semantic captcha on handset devices, Multimedia Tools and Applications, vol. 74, pp. 5141–5156, 2013.

[10] Y. Feng, Q. Cao, H. Qi, and S. Ruoti, Sencaptcha: A mobile-first captcha using orientation sensors, Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, vol. 4, no. 2, 2020, pp. 1–26.

[11] M. Guerar, M. Migliardi, F. Palmieri, L. Verderame, and A. Merlo, Securing pin-based authentication in smartwatches with just two gestures, Concurrency and Computation: Practice and Experience, vol. 32, no. 18, p. e5549, 2020.

[12] M. Guerar, L. Verderame, A. Merlo, F. Palmieri, M. Migliardi, and L. Vallerini, Circlepin: A novel authentication mechanism for smartwatches to prevent unauthorized access to iot devices, ACM Trans. Cyber-Phys. Syst., vol. 4, no. 3, Mar. 2020.

[13] M. Guerar, B. Mohamed, and V. Alimi, Color wheel pin: Usable and resilient atm authentication, Journal of High Speed Networks, vol. 22, pp. 231–240, June 2016.

[14] S. Rees-Pullman, Is credential stuffing the new phishing?, Computer Fraud & Security, vol. 2020, no. 7, pp. 16 – 19, 2020.

[15] M. Guerar, L. Verderame, M. Migliardi, and A. Merlo, 2gesturepin: Securing pin-based authentication on smartwatches, 2019 IEEE 28th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2019, pp. 327– 333.

[16] T. Hupperich, K. Krombholz, and T. Holz, Sensor captchas: On the usability of instrumenting hardware sensors to prove liveliness, Trust and Trustworthy Computing, M. Franz and P. Papadimitratos, Eds. Cham: Springer International Publishing, 2016, pp. 40–59.

[17] V. C. Mantri and P. Mehrotra, User authentication based on physical movement information, Patent 9 864 854, January, 2018.

[18] Brave. (2019) zksense: a privacy-preserving mechanism for bot detection in mobile devices. [Online]. Available: https://brave.com/zksense-a-privacy-preserving-mechani- for-bot-detection-in-mobile-devices/

[19] D. Asonov, and R. Agrawal, Keyboard acoustic emanations, IEEE Symposium on Security and Privacy. Proceedings. 2004, Berkeley, CA, USA, 2004, pp. 3-11.

[20] I. Shumailov, L. Simon, J. Yan, and R. Anderson, Hearing your touch: A new acoustic side channel on smartphones, arXiv, 2019.

[21] J. V. Monaco, SoK: Keylogging Side Channels, 2018 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 2018, pp. 211-228.

[22] S. Chen, R. Wang, X. Wang and K. Zhang, "Side-Channel Leaks in Web Applications: A Reality Today, a Challenge Tomorrow," 2010 IEEE Symposium on Security and Privacy, Berkeley/Oakland, CA, 2010, pp. 191-206.

[23] Guerar, M., Verderame, L., Migliardi, M., Palmieri, F., & Merlo, A. (2021). Gotta CAPTCHA 'Em All: A Survey of Twenty years of the Human-or-Computer Dilemma. ArXiv, abs/2103.01748.