I Am (Still) Here!

Inês Albano
INESC-ID
Lisboa, Portugal
ines.albano@tecnico.ulisboa.pt

Daniel Gonçalves
INESC-ID
Lisboa, Portugal
daniel.goncalves@inesc-id.pt

***Abstract*—Students must be accounted for in a classroom environment, and there are several methods to ensure student checking. Traditionally, attendance can be done with paper-based solutions such as the signing of attendance sheets. However, this is unreliable, time-wasting, and not scalable. In this paper, we describe an automated, costless, portable, web-based platform that is resilient to fraud thanks to device fingerprinting techniques, retrieving indicators such as the user-agent, IP address, fonts installed, geolocation, and others.**

Various simulations tested our solution and proved that the platform could detect fraud using the fingerprinting mechanism and the geolocation method.

***Index Terms*—Student Attendance System, Automated Attendance System, Student, Professor, Attendance, Web Application, Device Fingerprinting, Geolocation.**

I. Introduction

To know which students attend the class and how many they are in a classroom environment, professors use several methods. The most widely used is calling a student's name or passing a sheet of paper for students to sign their name on. These methods involve **wasting class time** and, if the second practice is used, the professor is then required to collect all the signatures and **manually enter that information into a computer**, which is prone to fraudulent behavior and human error. There is little to no way to ensure a student does not sign a colleague's name or even ensure no errors occur when transcribing the physical information to digital format. Neither of the described methods above are scalable, reliable, or efficient, as it takes time to count all students and verify their identity.

Current research focuses on finding techniques to make this process more efficient and less fraudulent. Researchers developed some attendance-taking methods, but they require extra hardware. Despite being costly and requiring maintenance, these "automated" methods do not have the disadvantages that come with manual attendance. These techniques are brought by the authentication methods, allowing users to authenticate using knowledge of **specific information**, with ownership of a **physical object** or even with an **unique trait or characteristic** that allows proving the user's identity, [1]. One of the only features that an attacker cannot reproduce is a person's biometric information. However, for the efficient use of biometric data, an investment in biometric readers is required. So, instead of buying equipment and improving the resilience of a system, other techniques can complement cheaper systems. These techniques involve the usage of **device fingerprinting**.

Device fingerprinting is the combination of device attributes to create a unique signature to identify a specific device. This technique has not yet been used to complement attendance-taking methods; however, it is feasible to apply, as most students have a computer or a mobile device with them every day. In this paper, we describe the creation of an **attendance taking system that is efficient and resilient to fraud using device fingerprinting.**

We evaluated which device fingerprinting techniques could be used in this work context, and performed experimental evaluations to verify whether the employment of the chosen device fingerprinting methods and geolocation techniques needed improvement for the application to work efficiently.

II. State of the Art

We will present previous scientific works related to Automated Attendance Systems (AAS) to try to find an efficient solution for the problem of attendance checking using device fingerprinting to avoid fraudulent cases.

A. Attendance Systems

State of the art can be divided into several categories. Some systems use simple technologies like pen and paper (manual systems), while others use more advanced technologies, like card readers, cameras, or other technologies.

1) Manual Attendance System: A common technique to register students' attendance is by calling their names or making them sign their name on a sheet of paper [2]. The amount of extra time makes this technique unfeasible, tedious to use, and unable to find fraud at the moment.

2) Radio-Frequency Identification: Attendance systems were developed using Radio-Frequency Identification (RFID), which is a technique that uses radio frequencies to identify physical objects [3]. One of the advantages of this system is that it is easy to implement and inexpensive due to the low cost of RFID cards. One of the significant disadvantages is that it requires extra hardware and it is not entirely anti-cheat, as students can scan other colleagues cards [2].

978-1-6654-8343-8/21/$31.00 ©2021 IEEE

2021 International Conference on Graphics and Interaction (ICGI) | 978-1-6654-8343-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICGI54032.2021.9655288

3) Biometrics: Using facial recognition to analyze a classroom can improve the teaching methods and implement more intelligent evaluations. However, these methods require cameras with excellent resolution, which can be very costly [4]. Unlike facial recognition, capturing students' fingerprints to validate attendance during classes and exams can also be costly, and the system needs fingerprint scans, passport photos, identification numbers, and user type to complete the user's profile for this approach to work [5].

4) Mobile and GPS Location: As the use of devices such as smartphones and laptops has become widespread across society, research was made to find cheap and viable solutions for student attendance with the help of those devices. Multifactor analytics can be used when registering attendance in a classroom: (1) a QR code shared by the teacher for the students to scan with their mobile phone, (2) the International Mobile Equipment Identity (IMEI) code saved on the database checks whether one of the phones is being used at that moment, (3) geolocation to ensure they are inside the same place as the other students, which works by checking the last three digits of the most common coordinates, i.e., the pair (latitude, longitude) is a practical solution that does not need the incorporation of extra hardware [6].

5) Software-only Systems: Several software systems allow for the confirmation of attendance using only a mobile device. Some systems use a solution based on the usage of Quick Response (QR) codes to register students' attendance right at the beginning of the class, intending to eliminate false attendances [7] [8]. There are also systems where students require a code before using it, such as Kahoot! [9]. Another software, Top-Hat [1] consists of the user submitting a four-digit code that the professor presents in the classroom and uses geolocation. However, students need to have their applications turned on during the whole duration of the class.

6) I Am Here![10]: is a system to take attendance efficiently while being resistant to fraud using device fingerprinting, which consists of the user inputting a series of random codes with letters and numbers to register their attendance in the system, after scanning a QR code or entering a link generated for that attendance. The platform checked if the browser and the device were the same as the previously used to try and combat fraud. Although the system has an efficiency of code insertions of 82.31%, the implemented device fingerprinting techniques were elementary and needed improvement.

B. Device Fingerprinting

The usage of device fingerprinting can be a means to combat fraud by associating a device with a user [11]. These mechanisms are divided into three categories: (1) Hardware related, (2) Browser related, and (3) Cross-browser.

1) Hardware related: The device's hardware features are more challenging to spoof than information extracted directly from the browser [12], as they can run in the background, leaving no trace behind and running without the user's awareness or consent, e.g., techniques based on clock-skew and network packet irregularities [13].

2) Browser related: Browser-specific fingerprinting uses Flash, Java, or JavaScript to extract relevant information, like User-Agent, Hypertext Transfer Protocol (HTTP) headers, screen resolution, installed fonts, plugins, browser platform, etc. [14]. When developers create extensions to protect the privacy of the user, they distinguish the device further from others [15], as the lack of available information distinguishes those clients further.

The systems that use Cascading Style Sheet (CSS) can gather fonts, screen information, and browser information, or using **Canvas** fingerprinting, recognizing a user based on the encoding of a hash of a Portable Network Graphics (PNG) image containing the contents of the canvas obtained using HyperText Markup Language (HTML)5 [16]. These techniques can be further combined with other technologies such as JavaScript, cookies, and canvas fingerprinting, using Cyclic Redundancy Check checksum, improving the identification of fingerprints that have already been recognized [17].

3) Cross-browser: The previous fingerprinting techniques do not enable the identification of a single user across different devices. Nevertheless, there is the possibility to identify several devices from the same home network [18]. Still, using different touch size devices to identify the same user in various devices is possible, with the combination of typing location accuracy, accelerometer on tap, and tapping duration. However, for this technique to work, the screen sizes need to be similar, and mobile devices cannot be compared with computer screen devices [19].

C. Discussion

Several systems approach the problem of attendance taking. However, none of them gives us the flexibility we desire. We will build a web platform that enables us to have a system that different devices can use and have the versatility we aspire. We desired to use both smartphones and computers to register the attendance, and for this, we cannot rely on the usage of the camera of the equipment, nor Bluetooth technology, since not all devices have these attributes. The completeness of the system is essential and can be achieved with device fingerprinting, specifically through browser device fingerprinting, since we cannot gather hardware information directly from the browser. Since our solution will need the client to send information, data needs to be taken with caution. Yet, considering that this is not a commercial system or a critical system, the techniques employed need to be reliable and resilient to fraud but simple to implement. In addition to the fingerprinting technique used, geolocation adds another layer of confirmation to the attendance.

III. Fingerprint Data

To create a fingerprint, we will use the combination of different pieces of information, like language, color depth, device

[1] https://support.tophat.com/s/article/Student-Attendance accessed 1st October 2020

memory, hardware concurrency, screen resolution, available screen resolution, timezone, platform, plugins, fonts, etc., gathered using FingerprintJS[2]. It is not possible to gather CPU information directly from the browser; however, it is possible to know the number of CPU cores. It is also not possible to create a fingerprint using biometric data, as not all devices have a physical biometric reader, e.g., fingerprint reader, nor is it possible to analyze typing behavior as biometric data because the typing patterns change based on the screen's size.

Despite this, to improve the identification accuracy, we will extract more information, such as the GPU model, whether the device is running an ad-blocker, and gather information about the device location.

A. Assessing Weights For Fingerprint Data

Some device information can be easily forged by clients, like a client's IP. To mitigate this problem, we can give different weights to different components of the fingerprint, giving less weight to components that can be easily spoofed and more weight to the more reliable ones.

To evaluate the user's authenticity registering the attendance, We calculate a weighted average of the retrieved indicators. First, we define the value for each indicator retrieved from the browser. After, we create an average weight for each user for each indicator based on the variation in fingerprint data from previous attendance records and finally normalize that value. The average weight is calculated by:

1) Evaluating whether an indicator changed comparing to the previous attendance session. If the indicator changes, the indicator in the current session has the pre-defined weight. Otherwise, the indicator has a weight of 0. We call the weight of the indicator in the current session as W.
2) We then apply W to Equation (1). Where $AvgWeight_{Ind}Session_N$ is the averaged weight for a specific indicator Ind for a certain session N, and the $AvgWeight_{Ind}Session_{N-1}$ is the same for the previous session of N.

$$\text{AvgWeight}_{Ind}Session_N = \frac{AvgWeight_{Ind}Session_{N-1}+W}{2} \quad (1)$$

3) The lower the average weight for an indicator, the most trustworthy it is because the less a variable changes, the closer to zero it becomes. Considering this, the user appears in the system for a specific attendance as one of three categories, represented with colors (green, yellow and red): **trusted**, if the number of indicators that are considered trustworthy is over that a pre-established number; **uncertain**, if the number of indicators that are considered trustworthy is not enough to establish the authenticity of the device; **fraudulent**, if the number of indicators that are considered trustworthy is less than the pre-established number to trust the device, and consequently the user's identity.

[2]https://github.com/fingerprintjs/fingerprintjs accessed 23rd December 2020

B. Finding the default values for each indicator

We based the default indicator value on the number of times they change, for example, due to updates, their possibility of being spoofed, and their relationship. We consider that all the indicators impact the fingerprint, and for simplicity reasons, we consider the value can vary from 1 to 5.

We divided the indicators into two categories: identify the device that changes over time and the information that is harder to spoof/change.

In the first category, the indicators session storage, local storage are indicators that can complete the fingerprint; however, they do not provide crucial information about the device, before we could understand that a user was or not using incognito mode, now we can only see if a user has those indicators enabled or not, which we will give a value of 1. The session storage, local storage, plugins, language, user-agent, screen resolution, available screen resolution indicators can change for various reasons, for example, navigator updates, installation of new plugins, not maximizing a navigator window, among other reasons, and for the reason that these parameters can change without user interaction, we will give these indicators the value of 2. For IP, platform, fonts, and ad-blocker, we will provide a value of 3, as these parameters can also change but do not change as frequently as the indicators explicitly before.

The second category focus on hardware characteristics of the device, which do not change as often, and these indicators will have a value higher than 3. The indicators audio, pixel depth, color depth, hardware concurrency, and device memory are numeral values, and the GPU is a nominal value, which can show complete information. With this in mind, the indicator GPU will be given a default value of 5, and the other indicators in this category will have a value of 4.

C. Calculating the fingerprint confidence

To achieve confidence for the fingerprint and the color for the *fingerprint* symbol, the confidence value is calculated using the technique proposed in III-A. Then the value is normalized, which is calculated by getting the value for each indicator of the fingerprint, through the formula in Equation (1), then sum all the values for the indicators, which the user can define their weight, and then normalize this value, through Equation (2), where the max value of the normalization is the sum of the original value of the indicators, and min is the minimum value that the indicators can reach, which in our case is zero. The minimum value is zero because it is the sum of the lowest values that the indicators can have, meaning that all the indicators can have the lowest value of zero.

$$fingerprint_{confidence} = \frac{sum_{indicators} - min}{max - min} \quad (2)$$

To understand why the fingerprint symbol in the overview of the class for that student has a specific color, the professor can click on the row associated with that student, as can be seen in fig. 1, which will lead the user to a new page.

The student's fingerprint information is divided into two sections on the new page: the fingerprint confidence throughout

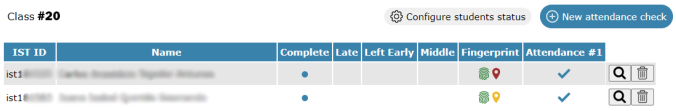

Fig. 1. Students that attended class number 20

the attendances for that course, which has a normalized value, and the confidence for each indicator for each attendance.

The indicators' colors are calculated similarly to the overall fingerprint confidence. However, the color associated with the indicator has to do with the closeness/furtherance to the original pre-defined weight for that indicator. The minimum value an indicator can have is zero, and the maximum is the initial value. There will now be six colors for each indicator, grey (undefined), dark green, light green, yellow, orange, red. The colors range from the most trustworthy to the less reliable, which can be seen in fig. 2.

A user can hover over the indicator and see the value for that indicator for that attendance and easily compare between sessions and understand why the indicator has that color. The user can also quickly know if the user is using a mobile device or a computer because, for each attendance, there is a symbol of a mobile phone or a portable computer associated. And, the fingerprint comparison happens only between the same devices, meaning only between mobile devices or computers.

In addition to employing fingerprinting techniques, the aim is to add geolocalization methods to ensure that the user is in the place where the attendance is taking place, allowing for a confirmation of the student's attendance.

IV. Geolocation

We will also collect their geolocation, which will be done using the geolocation API available in all modern web browsers after the users provide their consent, using *navigator.geolocation*[3]. The API will give us a user's location, estimated location of the IP address, or even GPS coordinates, if available in the device being used. The location will be provided *(latitude, longitude)* pair, decimal point numbers with, typically, up to 5 decimal places. This amount of decimal places gives us the user location with a theoretical precision of up to 1 meter at the equator. However, the decimal precision varies depending on the browser at use. Nevertheless, this precision can be much lower in real-life settings, mostly indoors (as is the particular case of attendance-taking). This imprecision hints that not all decimal places may be usable in our context.

The geolocation API returns the location's accuracy in meters, making it necessary only to consider the decimal places that satisfy the accuracy returned. If the accuracy is 1 Km, only two decimal places will be sufficient to represent the location, as the rest of the decimal places are not relevant for the location's accuracy. Otherwise, it would be more accurate (have a lower value).

[3]https://developer.mozilla.org/pt-BR/docs/Web/API/Navigator/geolocation accessed 6th May 2021

As geolocation is one of the most **privacy-sensitive** data we can use, we will not send it to the server. Instead, we will compute the hashes of each of the coordinates and send and store those hashes. We will use the SHA-256 algorithm, as it is secure since it is infeasible to reverse the hashing process, and collisions are unlikely.

To reduce errors induced by the GPS while still allow the verification of "close enough" locations, we compute **several hashes** at different accuracy levels and give the different hashes an **accuracy level** for the precision they hold. We will hash the coordinates up to 5 decimal places and store the browser's accuracy value (in meters). We will also create hashes for the coordinate with five decimal places (1.11 m), holding an accuracy level of 5, with four decimal places (11.1 m), an accuracy level of 4, with three decimal places (111 m), an accuracy level of 3, with two decimal places (1.11 Km), an accuracy level of 2, and with one decimal place (11.1 km), with an accuracy level of 1. The higher the accuracy, the better we can pinpoint where the user is.

Understanding if a user is at a particular location when recording attendance becomes a matter of comparing hashes. If they are the same, so will, most likely, be the locations. We will compare the student's hashes with the most common hashes for that class. Based on the accuracy of the student's location and where the most common hashes differ from the user's hashes, the confidence for that geolocation will be given.

A. Calculating the geolocation confidence

Similar to the fingerprint, the geolocation will also be associated with a color (dark green, yellow or red). This feature is only helpful in a traditional class context, as students' location can be compared to assure that students are within a certain radius. If they are less or equal to 10 meters of the radius, the color of the pin is green. If they are more than 1 Km away from most of their classmates, the color will be red; otherwise, it is yellow. It is essential to understand that when the browser gathers the geolocation, there is an accuracy associated with it, which means some students cannot pinpoint a specific location. The pin will only be green if the accuracy is less than 10 meters; otherwise, there is insufficient information to put that student inside the radius. Each level of confidence for the geolocation is associated with normalized values (0, 0.2, 0.4, 0.6, 0.8, 1) used to calculate the final fingerprint confidence.

When a row corresponding to a student is clicked, fig. 1, the user will be able to understand the evolution of the confidence of the geolocation confidence in the row with the name *Geolocation*, that when each square is hovered, it will show the accuracy gathered by the browser and the normalized confidence for each attendance until and including that class, an example is visible in fig. 2.

V. Evaluation

To analyze the best mechanism to detect fraud, in total, we implemented four algorithms to understand if the proposed mechanism in Section III-A is the best option. Each algorithm

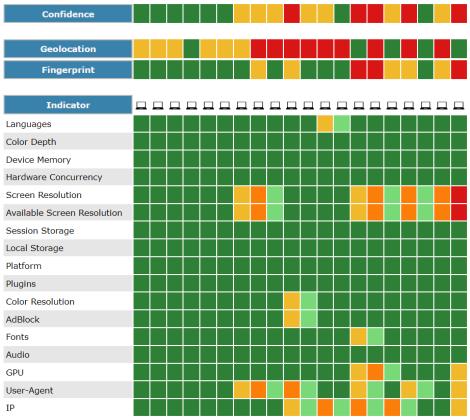

Fig. 2. Overview of student X's confidence, with the combination of fingerprint and geolocation. Each column corresponds to an attendance check, each row corresponds to the confidence for each indication of the fingerprint.

calculates the combination of indicators to create a fingerprint differently:

- **Algorithm 1** - The first algorithm we present is the one proposed in Section III-A. After getting the indicators for each class, we will compare them with the previous class, and if the indicator changes from one class to the other, we will add a fixed value to the previous value and divide it by two. If the final value moves towards zero, an indicator is more trustworthy; otherwise, it is not.
- **Algorithm 2** - The second algorithm is an alternative version to the previous algorithm. Instead of summing the whole fixed value for the indicator, when the indicator changes, it sums only part of the fixed value of the indicator, based on where it last detected that indicator. With this in mind, in the formula Equation (1), W, will the calculated in the following manner:
  - If the indicator changed and there was a registered value for that indicator equal to current one, W will be calculated by eq. (3), where $current_{index}$ is the index of the indicator in the current session, and $last_{seen\_index}$ is the last session where the indicator value equal to the current session was seen.

    $$\text{W} = \text{Predefined}_{weight} * \frac{current_{index} - last_{seen\_index}}{current_{index}} \quad (3)$$

  - If the indicator changed, but there is no equal value registered, W will have the pre-defined weight value.

  This will improve the previous algorithm if the user has two of the same devices, such as two portable computers, and changes the device every time they register attendance. With the previous algorithm, the final calculated value would always move towards the maximum value possible.
- **Algorithm 3** - The third algorithm implemented consisted of only comparing the registered value of the indicator with the previous value. This algorithm only has two states: equal to the last value (green) or different from the previous value (red).
- **Algorithm 4** - The last implementation was completed by comparing the indicator value with the most common registered until that class. Again, this only has two states: equal to the most frequent indicator (green) or different from the most frequent indicator (red).

A. Fingerprinting algorithm comparison

For each algorithm, we applied the fingerprint of a student from the semester of 2020/2021, meaning the data for the fingerprint is the same, and the weight for the indicators is the same for Algorithms 1 and 2.

Considering the evolution of the student's fingerprint, and looking at the indicator **GPU**, for the 14th attendance for a class, the indicator has the value of *Intel(R) HD Graphics 530*, which is an integrated GPU, however for the previous attendance, the registered value for the GPU is *NVIDIA GeForce GTX 960M* for that attendance, which is a dedicated GPU. The reported GPU depends on the GPU that the system is currently using, which can depend on several factors, for example, the power settings of the device. With this in mind:

- We expected that for Algorithm 1, the indicator would become less trustworthy (yellow or orange). The platform shows the color corresponding to that attendance for that indicator is yellow, as it was expected since the confidence state previous to this one was dark green (very trustworthy). This algorithm shows that now that indicator is to be considered in a level of confidence of uncertainty.
- For Algorithm 2, we expect the indicator would become less trustworthy (yellow, orange, or red). As with the previous algorithm, the platform showed the color is yellow (uncertain). This is due to the algorithm not having registered that value for that indicator. However, looking at the end of the row for the GPU (for the last attendance). We could explicitly see the difference between the first and the second algorithm, as Algorithm 2 has a dark green color for the previous attendance, while Algorithm 1 has a yellow color since in the last attendance, it registered *Intel(R) HD Graphics 530*. Still, the last time it registered that value was in the attendance outline in black, making the confidence have a lower value (more trustworthy) in Algorithm 2 rather than in Algorithm 1.
- We anticipate that for Algorithm 3, the indicator would have a red color since it changed compared with the previous value. The platform showed that the confidence is untrustworthy (red), which is consistent with our hypothesis.
- And for the Algorithm 4, the indicator will also have a red color, since the most common value is *NVIDIA GeForce GTX 960M* and not *Intel(R) HD Graphics 530*. The platform outputs red, and we confirm our theory since the value registered in that attendance is not the same as the most common value.

B. Fingerprint algorithms discussion

With the previous statements in mind, we can make some conclusions about the several algorithms implemented:

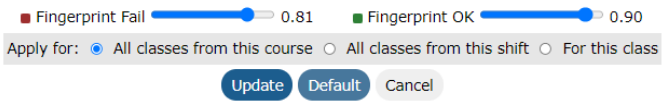

Fig. 3. Snippet of the interface that enables change of threshold for the algorithms.

- The Algorithm 3 and 4 are the worst options, as the confidence for the fingerprint was the lowest, even though the student was always the same. However, these are good algorithms to use if, for Algorithm 3, the professor wants to be warned if something changed about the student, and for Algorithm 4, if the professor is assured that the students only use one device. However, suppose a student changes devices, for example, changes their phone, by the end of the semester. In that case, the fingerprint will always be untrustworthy, not having the time to update the most common fingerprint values.
- The Algorithm 1 and 2 were the most complete, since they have the usage of weights of the indicators, since not all indicators have the same weight for the calculation of the fingerprint, meaning that if, for example, the user has a second monitor, that weight can be lower and have less impact in the final fingerprint if the indicator changes. However, choosing between one or the other depends on what the professor may want. If a professor wants to keep a more conservative route, the first algorithm is better since constant changes may trigger the system to believe that the student is at fault. Otherwise, if the professor wants to detect more prominent fraud cases, the second algorithm is better since it considers if the user has already registered a particular indicator, which means it accounts for a student having more than one monitor, for example.

Using **Algorithm 1**, we performed a simulation to understand how the fingerprint confidence value changed with different scenarios. We tested the algorithm for six different scenarios and explained what we expected would happen against what truly happened. The algorithm has a threshold value that reveals if the student is in fraud. This threshold has the default value of 0.810, which was gathered experimentally. However, this threshold can be configured by the user, as can be seen in fig. 3.

C. Fingerprint simulation with different scenarios

- **Scenario 1** - A student registers three attendances with the same device, not changing any characteristics about the registry. We expected the confidence value to stay at 1.000 because no changes were made to the fingerprint, which was proven when the platform shows the last row with all green indicators.
- **Scenario 2** - A student registers three attendances with the same device but with a different browser. By changing the browser, other indicators may vary depending on the browser, for example, the plugins. In our case, we first user Google Chrome and Microsoft Edge for the last attendance. We expected the value for the confidence to lower but not to impact the final color of the fingerprint (staying green). We were able to see that the language, plugins, and user-agent indicators changed, leaving the confidence with a value of 0.940, not changing the color of the fingerprint, which is what we expected.
- **Scenario 3** - We wanted to test what would happen if a user kept using the same device but changed the network. For this, we performed two attendances with WIFI on a personal network, and then the last attendance, we used a VPN. We expected none of the indicators to change, apart from the IP address, and the platform output the expected result the result.
- **Scenario 4** - To ensure that while using mobile data, the same occurrence would happen as in *Scenario 3*, we performed two attendances with the same device using WIFI in a private network, and the last attendance was registered using the same device using mobile data. We expected that only the IP address would change, which happened as the platform output the result.
- **Scenario 5** - This scenario was achieved by performing two attendances using a computer and the last attendance using another computer. We expect the program to detect a fraudulent case. The platform showed that several indicators changed, leaving the confidence for the fingerprint with a value of 0.670 and a red color, which was what we expected.
- **Scenario 6** - The last scenario was equal to the one stated previously. However, using a mobile device instead of a computer. We performed the first two attendances using a mobile device and then changed it to another device to register the last attendance. We would expect the same to happen as to *Scenario 5*. The platform showed that the final calculation of the confidence for the fingerprint decreased the value from 1.000 to 0.800; as the threshold was at 0.810, it would have a red color.

D. Fingerprint simulation discussion

We used different algorithms to compare the methods of combining the indicators gathered from the users. After using the algorithm proposed in III-A, we deducted that the algorithm behaved how we expected for all the different scenarios that could happen in a regular class.

Our goal of finding indicators that identified a user's device was satisfied. We can see the difference between fingerprints when we explore their fingerprint and by the value of confidence.

E. Geolocation simulation

Due to the pandemic, we could not test the geolocation algorithm in real-time. However, we performed a simulation with different scenarios to make conclusions about the performance of this algorithm.

We used Google Maps to gather the different coordinates that we used to test the algorithm. Location 1, 2, 3 and

4, have the coordinates (38.737129, 9.302584), (38.737080, 9.302573), (38.736502, 9.302143), and (38.6847442, -9.3140146), respectively.

We have five different scenarios, where we explain what we expected would happen when we employ the method proposed in IV, and what is the output of the system, in each scenario.

There are six students (A, B, C, D, E, F) taking attendance for a class, and the confidence (represented by the square) is only calculated with the confidence of the geolocation and not the combination of fingerprint-geolocation:

- **Scenario 1** - *Four students are in location 1, two students are in location 2.* We expect that all students have a confidence relative to the geolocation of 1.0 because only the two last digits of the coordinates are different. *The platform shows students A, B, C, D, E, F have green color for the geolocation.* We can see that all students have a green status, and all have a confidence value for the geolocation of 1.0, which agrees with our hypothesis.
- **Scenario 2** - For this scenario, there were *five students in location 1, and one is in the restaurant in the university*. We expect that the five students in the same place have confidence relative to the geolocation of 1.0, and the one in the restaurant has a lower confidence value. *The platform shows students A, B, C, E, F have green color for the geolocation, while D has a yellow color.* All five students have a green status (confidence of geolocation 1.0), and the sixth student (D) has a yellow color, which, when clicked on, shows the confidence for the geolocation of 0.4, meaning the coordinates were not the same from the third decimal place forward, which is in agreement with our hypothesis.
- **Scenario 3** - In this scenario, *there were four students in location 1, one (B) is in location 2, and the last of the six (D) is in the restaurant in the university*. The platform shows students A, B, C, E, F have green color for the geolocation, while D has a yellow color. We expect the four students and student B to have a green color for the square, and student D has a yellow color since it is still inside the university. *The platform shows students A, C, E, F have green color for the geolocation, while B and D have a yellow color.* We see that students A, C, E, F have a green status (confidence of geolocation 1.0), and B and D have a yellow color, which is not what we expected. However, in further analysis, we see that student B has a confidence for the geolocation of 0.6 and student D only has a value of 0.4. Student B failed on the fourth decimal place of the longitude, whereas student D failed on the third decimal place of the longitude. Also, both students B and D are separate. If there were another student near B, we would have a scenario similar to Scenario 1, where D would still have a yellow color.
- **Scenario 4** - The fourth scenario, we have *four students in location 1, one (B) is in location 3, and one (D) in location 4*, which is about 7 kilometers away from the university. We expect the four students in location 1 have a green color for the square, student B to have a yellow color, and student D to red. *The platform shows students A, C, E, F have green color for the geolocation, while B has a yellow color, and D has a red color.* We see that students A, C, E, F have a green status (confidence of geolocation 1.0), and B has a yellow color, with a confidence of 0.6. Student D, with red color, and confidence of 0, since the first decimal place of the longitude, is different from the most common first decimal place of the longitude, which is what we expected.
- **Scenario 5** - The last scenario represents a flaw of this algorithm, we have four students in location 4 (near the beach, about 7 kilometers away from the university), and two (B, D) are in location 1. We expect that the four students in location 4 have a green color for the square, students B and D have a red color. *The platform shows students A, C, E, F have green color for the geolocation, while B and D have a red color.* We see that students A, C, E, F have a green status (confidence of geolocation 1.0), and student B and student D, with red color, and confidence of 0, since the first decimal place of the longitude is different from the most common first decimal place of the longitude, which is what we expected. As most of the students in this scenario gathered near the beach, instead of a classroom to register their attendance, the system assumes that the most users are where the class is taking place.

F. Geolocation conclusion

Based on the simulations, we concluded that the geolocation mechanism works if most of the students who are not performing fraud are in the same place. However, we assume that most of the students will be in class at the moment of attendance registration, which means there will be a set of students whose coordinates are going to the classroom. However, as a future improvement, we could complete our geolocation mechanism by gathering the coordinates of the professor when they start the attendance taking and then use those coordinates to compare which the most common coordinates of the students to ensure that the most common coordinates are indeed in the same place as the professor, which would solve the problem of Scenario 5. Although this method seems to be the most complete, we have to be reminded that this a web application, and the accuracy of the coordinates depend on the device the user has, which means if the device the professor is using has a high inaccuracy, the professor's coordinates might not be useful to understand if the most common coordinates are indeed in the same place as the professor is.

VI. Conclusion

We proposed a system that made it possible to register a student's attendance for a class and detect fraudulent cases using fingerprinting and geolocation mechanisms, with the advantages of being *costless* and *portable*. We improved on an existing cross-platform system and implemented new features. The new system *I Am (Still) Here!* answers the

problem authenticity of the student register their attendance. Not only were we able to register students' attendance using a simple and effective platform, but our implementation also covers the most fraudulent cases we discussed previously, providing an efficient solution for the professor to find if a student is performing in a fraudulent manner while providing customization for the user, allowing each professor to tweak the outcome of the system to their liking.

VII. System Limitations and Future Work

A limitation of our system is the accuracy of the geolocation. To improve this, we would need to have the usage of devices that had GPS incorporated. With this in mind, our solution would be to build Android and iOS applications that would use the GPS directly from the device and not the geolocation from the browser at use.

Finally, to improve the fraud detection using device fingerprinting, we could implement machine learning algorithms that would learn with the number of classes registered and evaluate the indicator weights for each user instead of the system.

Acknowledgment

This work was supported in part by the National Funds through the Fundação para a Ciência e a Tecnologia (FCT) under Project UIDB/50021/2020, and in part by the Project GameCourse, Portugal, under Grant PTDC/CCI-CIF/30754/2017.

References

[1] S. N. Abdulkader, A. Atia, and M.-S. M. Mostafa, "Authentication systems: Principles and threats," *Computer and Information Science*, vol. 8, no. 3, p. 155, 2015.

[2] D. A. Scanlan, "An inexpensive rfid attendance system," *J. Comput. Sci. Coll.*, vol. 25, no. 2, pp. 19–29, Dec. 2009, ISSN: 1937-4771.

[3] R. Gujral, "Any time any where-remote monitoring of attendance system based on rfid using gsm network," *International Journal of Computer Applications*, vol. 39, pp. 37–41, Feb. 2012.

[4] C. Ping, H. Da-Peng, and L. Zu-Ying, "Automatic attendance face recognition for real classroom environments," in *Proceedings of the 2018 2nd International Conference on Big Data and Internet of Things*, ser. BDIOT 2018, Beijing, China: Association for Computing Machinery, 2018, pp. 65–70.

[5] U. E. Peter, C. K. A. Joe-Uzuegbu, L. Uzoechi, and F. K. Opara, "Biometric-based attendance system with remote real-time monitoring for tertiary institutions in developing countries," in *2013 IEEE International Conference on Emerging Sustainable Technologies for Power ICT in a Developing Society (NIGERCON)*, Nov. 2013, pp. 1–8.

[6] S. Tachmammedov, Y. K. Hooi, and K. S. Kalid, "Automated multi-factor analytics for cheat-proofing attendance-taking," in *Proceedings of the 2018 7th International Conference on Software and Computer Applications*, ser. ICSCA 2018, Kuantan, Malaysia: Association for Computing Machinery, 2018, pp. 183–188.

[7] F. Almasalha and N. Hirzallah, "A students attendance system using qr code," *International Journal of Advanced Computer Science and Applications*, vol. 5, Jan. 2014. DOI: 10.14569/IJACSA.2014.050310.

[8] Z. Ayop, C. Yee, S. Anawar, E. Hamid, and M. Syahrul, "Location-aware event attendance system using qr code and gps technology," *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 9, pp. 466–473, 2018.

[9] R. Asa'd and C. Gunn, "Improving problem solving skills in introductory physics using kahoot!" *Physics Education*, vol. 53, no. 5, p. 053 001, Jun. 2018.

[10] M. Nascimento and D. Gonçalves, "I am here!" In *2019 International Conference on Graphics and Interaction (ICGI)*, Nov. 2019, pp. 8–15.

[11] P. R. Yogesh and S. R. Devane, "Primordial fingerprinting techniques from the perspective of digital forensic requirements," in *2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, Jul. 2018, pp. 1–6.

[12] G. Nakibly, G. Shelef, and S. Yudilevich, *Hardware fingerprinting using html5*, 2015. arXiv: 1503.01408 [cs.CR].

[13] G. Acar, "Obfuscation for and against device fingerprinting," in *Proc. Symp. Obfuscation*, 2014.

[14] N. Nikiforakis, A. Kapravelos, W. Joosen, C. Kruegel, F. Piessens, and G. Vigna, "On the workings and current practices of web-based device fingerprinting," *IEEE Security Privacy*, vol. 12, no. 3, pp. 28–36, May 2014.

[15] R. Upathilake, Y. Li, and A. Matrawy, "A classification of web browser fingerprinting techniques," in *2015 7th International Conference on New Technologies, Mobility and Security (NTMS)*, Jul. 2015, pp. 1–5.

[16] N. Takei, T. Saito, K. Takasu, and T. Yamada, "Web browser fingerprinting using only cascading style sheets," in *2015 10th International Conference on Broadband and Wireless Computing, Communication and Applications (BWCCA)*, Nov. 2015, pp. 57–63.

[17] W. Jiang, X. Wang, X. Song, Q. Liu, and X. Liu, "Tracking your browser with high-performance browser fingerprint recognition model," *China Communications*, vol. 17, no. 3, pp. 168–175, Mar. 2020.

[18] C. Blakemore, J. Redol, and M. Correia, "Fingerprinting for web applications: From devices to related groups," in *2016 IEEE Trustcom/BigDataSE/ISPA*, Aug. 2016, pp. 144–151.

[19] H. Yuan, C. Maple, C. Chen, and T. Watson, "Cross-device tracking through identification of user typing behaviours," *Electronics Letters*, vol. 54, no. 15, pp. 957–959, 2018.