A Multi-layer Approach for Mobile Devices: Behavioral Fingerprinting

Gabriel-Cosmin APOSTOL[1], Alexandra-Elena MOCANU[1], Bogdan-Costel MOCANU[1]*, Dragoș RĂDULESCU[1], Cătălin NEGRU[1], Ionuț PETRE[2], Florin POP[1,2]

[1] University Politehnica of Bucharest, 313 Splaiul Independenței, Bucharest, 060042, Romania
gabriel.apostol@stud.acs.upb.ro, alexandra_elena.mihaita@cti.pub.ro, bogdan_costel.mocanu@upb.ro (**Corresponding author*), dragos.radulescu@stud.acs.upb.ro, catalin.negru@upb.ro, florin.pop@upb.ro

[2] National Institute for Research & Development in Informatics – ICI Bucharest, 8-10 Maresal Averescu Avenue, Bucharest, 011455, Romania
ionut.petre@ici.ro, florin.pop@ici.ro

Abstract: In distributed systems such as mobile networks, problems like heterogeneity and real-time reliable responses represent constant challenges. Furthermore, reliable identification of the participants in mobile networks is a challenging research problem because of the susceptibility to security attacks. Also, reliable identification of participants means to be able to distinguish between the device and its user. In this paper, a multi-layer mechanism is proposed for reliable identification of devices and their users in mobile networks, based on the smartphone sensor data. The proposed model can uniquely identify both the device and the users by leveraging a mobile application that harvests mobile sensor data in various usage scenarios and analyse different usage patterns. Moreover, in order to achieve data confidentiality, a cryptographic scheme is integrated, based on the public key exchange mechanism. The obtained results show that the model proposed in this paper provides meaningful insights about user behavioral patterns, without special permissions and without disclosing the actual owner identity or biometric data.

Keywords: Mobile fingerprinting, Security, Data protection, Mobile ad-hoc networks, MANET.

1. Introduction

Current Internet and web services were not engineered to maintain a high degree of privacy for the personal data that users share. In general companies, governmental or not, are interested in getting statistical data about the users who were connected at some time to a certain webpage or web service. Therefore, it is reasonable to consider that any network can be defined as a secure private space because the access is unregulated, and not all users have equal rights.

In the last decade, many people have become aware that the privacy and confidentiality of their data (Sicari, Rizzardi & Coen-Porisini, 2022) can be trespassed. Therefore, they began searching for alternatives. To communicate freely and securely, a group of people can decide to create a decentralized network that is controlled by them. An ad-hoc infrastructure can operate over multiple communication channels, like IrDa, NFC, Bluetooth, Ethernet, and Wireless, without depending on a regulatory entity, as mentioned in (Jayachandran & Jaekel, 2022). Due to this fact, this architecture is resilient to censorship, surveillance, and even failures.

According to Al-Shareeda & Manickam (2022), ad-hoc mobile networks are susceptible to serious attacks due to their spontaneous nature and due to the lack of access control mechanisms set in place, to prevent malicious actions. Since no authentication mechanism is used, identifying a certain device can be unfeasible, even when a subject is already connected. Another drawback of this approach resides in the fact that no authorization process takes place. Therefore, any peer could turn malicious at any moment, without anyone noticing it until it is too late.

The first step in designing a feasible solution with a reasonable amount of security is to create a verification mechanism to enhance the trust between the participating nodes. This represents a significant challenge since a very small amount of information could be considered trustworthy to generate a uniquely identifiable fingerprint of the device. Even if there are many hardware and software components uniquely identifying a device, there is no actual methodology to prove they are not forged, to impersonate another device.

Although a computed fingerprint cannot be entirely trusted, there are emerging technologies that promise to provide tamper-proof methods of data acquisition, transfer, and processing. Even though at the moment there are open problems regarding this area, the present research aims to find a reasonable solution for mobile device identification and authorization in ad-hoc networks, based on the data retrieved from the device's sensors (Figure 1).

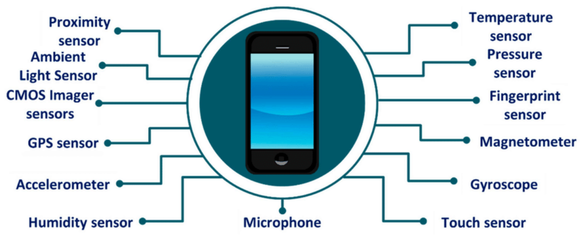

Figure 1. Possible sensors on smartphones (Majumder & Deen, 2019)

The novelty of the paper consists of a multilayer architecture for distributed mobile device fingerprinting which, to the best of our knowledge, has not been covered in this manner by other research papers.

Mobile device fingerprinting is a procedure that comprises methods for uniquely identifying a specific machine, to ensure authorization and communication between the participating nodes of a decentralized network. This network is operated collaboratively, rather than being controlled by a single entity as presented in (Ciobanu et al., 2017).

Regarding mobile devices, the spontaneous association between more nodes is referred to as a MANET (Mobile Ad-Hoc Network). Due to the unattended nature of such networks, a unique identifier must be generated automatically by the node itself, without the existence of a certified authority that could detect a malicious parameter of a device. Also, the privacy and confidentiality level of the transmitted messages is an open problem (Korir & Cheruiyot, 2022), which will be addressed by the research proposed in this paper.

Therefore, the main contributions of the present proposal are the research and design of a multilayer profiling architecture that offers increased security in mobile cloud applications through mobile phone sensor fingerprinting.

The structure of the paper is as follows. Section 1 presents an introduction to the topic of behavioral fingerprinting, Section 2 presents the related works in the field, Section 3 describes the proposed solution, Section 4 presents the experimental evaluation and, finally, Section 5 concludes the results of the paper.

2. Related Work

Device fingerprinting is a methodology based on standard or non-standard procedures (Sánchez Sánchez et al., 2023), used to uniquely identify a specific physical device in a network. The resulting fingerprint value is a metric that measures the anonymity of a browser, an operating system, or other specific hardware and software markers.

These procedures appeared due to certain categories of service providers which needed to identify a subscriber without the use of general information such as geo-location coordinates, IP or MAC address, etc., because these values can be easily spoofed or modified. Also, using virtual private networks, proxies, or Tor, an attacker could impersonate a valid peer, without the victim's knowledge. This validation procedure is currently used in banking, health applications, and other login-based web services, where identifying a malicious device is essential, even vital.

The mobile device fingerprinting procedure is similar to the above-mentioned ones because mobile terminals share similar hardware and software features with desktops or laptops. Therefore, a mobile browser could supply a User Agent or other specific data, to compute a fingerprint. However, this procedure could output two identical values for two physical devices produced by the same manufacturer.

To be considered valid, the authors Wei, Gu & Du (2022) argue that a unique identifier should meet the following requirements: have high entropy, so the fingerprint could be immediately associated with the device, to be persistent over time, and to be resilient to forgery. Also, it should be easy to integrate into a given architecture, facilitating connectivity and interoperability between solutions. Besides these requirements, the procedure should consider the following device fingerprinting characteristics (Fukumoto, Ano & Goto, 2013):

- No impact on the user experience or the infrastructure, because asking for specific hardware or software components could pose financial obligations for both users and network administrators. From a security standpoint, counterfeit versions of the software could contain back-doors or malicious code, which could infect the network or its participants. Also, if they require special hardware, these modules should be easily installable on any given device, without maintenance costs or additional infrastructure upgrades;

- A real-time decision module should be imlemented to quickly determine if a communication participant is malicious or not. As a consequence, the blacklisted users cannot interact with the legitimate ones and cannot disseminate malicious messages;

- Fuzzy matching implies computing separate hash values for each verifiable piece of information. If the fingerprint procedure is browser-based, hashing all the required marker values would produce different results on the same machine. Therefore, maintaining separate hash values increases the degree of verifiability.

Given the diversity of hardware and software components and capabilities, the fingerprinting procedure could be designed in the following manner (Alaca & van Oorschot, 2016):

- Client-based. This approach implies that the computation is made locally, to gain access to all kinds of resource details, like hard-disk serial numbers, network card MAC addresses, or other identifiers obtainable by a simple user. The resulting value should be unique, persistent, and tamper-proof;

- Server-based. Unlike the above-mentioned method, this procedure implies a profiling server, responsible for collecting the user's browser data, operating system, and network connection details. The fingerprint value is computed on the server, without the need to install additional client-side software on the given workstation or personal computer. This approach is mainly used in online banking and e-commerce websites;

- Browser-based. This approach relies on data collected using the user's browser, like cookies, user agent value, generic and privacy settings, installed plug-ins, etc. Because the computation is made locally, without installing any additional software, this procedure is used very frequently;

- HTTP fingerprinting-based. This method consists in extracting specific metadata transmitted through an HTTP connection. Some of the used parameters are the compression method name, the support for proxies, the supported cipher suite, etc.;

- Operating system-based. This approach implies computing the unique identifiable value using the operating system information obtained from a network connection with the physical device;

- TCP-based. Similar to the HTTP-based approach, this method relies on TCP protocol stack information gathered when a network connection is established.

However, using a forged signature, a malicious participant could use this data to initiate various attacks over the network. The most frequent types of threats are (Benson et al, 2015):

- DoS (Denial of Service) attack. The attacker initiates multiple actions aiming at its victim, to overload it or to get blacklisted;

- Impersonating a legit participant. This attack implies using the fingerprint associated to initiate actions or transactions with stolen credentials;

- Eavesdropping. An adversary uses the extracted data to passively filter the network traffic and gather more information about his victim;

- Back door of pre-embedded procedures. This attack relies on the existence of misconfigured pre-embedded procedures, to obtain access to the application source code to control the execution flow and maintain access.

The authors Hupperich et al. (2015) also identify five strategies that can be used to detect possible fraud or cyber-attacks over the fingerprint procedure:

- Anomaly detection. It identifies the configurations and network-wide anomalies, such as using proxy servers or the Tor client;

- Device velocity. Velocity filter can be used to minimize the success rate of cyber-attacks or fraud campaigns;

- Transaction linking. The fingerprint value can be used to identify the entity which initiated a specific action or transaction;

- Account linking. It identifies if the account which is associated with a certain fingerprint was accessed in an unauthorized manner;

- Device reputation. The fingerprint associated with previously identified malicious users is blacklisted, to protect other participants.

To generate a unique identifier for a node, it is necessary to find a set of spoofing-resistant verifiable hardware and software attributes. As regard to mobile devices, there is some information that is hardly changeable, like the resolution of the screen, the hardware sensors list, the device manufacturer, and the operating system name. Some attributes can be obtained through non-elevated software API calls, and others require a high-security privilege context, which could pose a privacy and security risk (Kurtz et al., 2016).

The main target of the present research is to leverage the unprivileged API calls to reliably compute a fingerprint which could determine the uniqueness of a node, because using highly privileged system calls is necessary to root or jailbreak the mobile device. The associated security risk of a highly privileged environment is given by the possibility that an attacker could execute instructions, without user authorization, which is an unacceptable trade-off between security and usability.

3. Proposed Solution

This section presents a multilayer architecture for fingerprinting in mobile distributed networks. The present proposal includes 3 layers as follows: underlay infrastructure and data sources; hybrid processing layer and application layer as shown in Figure 2.

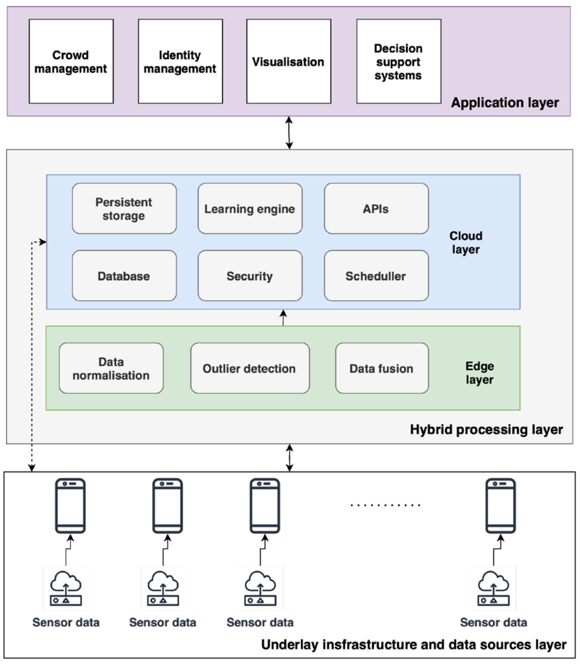

Figure 2. Overview of the multilayer approach for security enhancement of mobile systems

Firstly, the underlay infrastructure and sensor data layer are responsible for data collection from the mobile sensors (proximity and light sensor, magnetometer, barometer, accelerometer, GeoMagnetic, and Gyroscope) and the communication and verification protocol. Furthermore, the hybrid processing layer is composed of two computation sublayers: the edge layer and the cloud layer. The key components of the edge layer consist of basic operations such as sensor data normalization, outlier detection, and data fusion. On the other hand, the cloud layer is responsible for persistent storage, a learning engine that includes support vector machines and neural network kernels, a security module responsible for data anonymization and confidentiality, a task scheduler for optimal processing, and external APIs. Thirdly, the application layer consumes the processed data from the hybrid processing layer through the external API. In the application, the layer can be implemented by multiple applications suitable for different use cases such as crowd management (Darsena et al., 2022), identity management (Zukarnain, Muneer & Ab Aziz, 2022), data visualization, and decision support systems (Li et al., 2021).

3.1 Android Devices Fingerprinting

A significant set of features can be collected from mobile device sensors. Mobile device fingerprinting is a technique used to identify and collect information about mobile devices. Thus, mobile device fingerprinting can be used to improve security engagement by providing a way to identify and track suspicious behavior on mobile devices and users.

Moreover, several aspects need to be analyzed. Taking into consideration the heterogeneity of the sensors, before any data computation, a normalization step and the fact that a device can be used by multiple users must be considered. The normalization of sensor data is a crucial step to extract context information. This is because different sensors may possess varying characteristics and sensitivities, leading to data that may not be directly comparable without normalization. Some other factors affecting sensors are temperature changes, aging of the sensor, and mechanical stress. To normalize data across different sensors, the infinite norm can be used, as shown in Equation 1, where *S* is a set of data from a sensor and *nS* is the normalized set of data.

$$\|S\|_{\infty} = max(S)$$
$$nS = S / \|S\|_{\infty} \quad (1)$$

Another aspect that needs to be taken into consideration is the line of sight (LOS) limitations. It is assumed that the underlying transport is backed by Bluetooth Low Energy. According to authors Katila, Gia & Westerlund (2022), it exhibits a maximum data transfer rate of 2 Mbps, which may vary, depending on the environment conditions, hardware, power constraints, and the modulation of signal (2MFSK, GFSK, 2GFSK). BLE signals can be weakened as they pass through obstacles such as walls, furniture, and other objects, and can reflect off surfaces such as walls, ceilings, and floors, causing multipath interference. Additionally, certain materials such as metal and water can absorb BLE signals, further reducing signal strength. LOS can help to reduce the number of obstacles, surfaces, and materials that the signal has to pass through, thereby improving signal strength and reducing interference. However, BLE is designed to be more resilient to signal attenuation, reflections, and absorption than traditional Bluetooth.

To determine a unique id for the Android device, it is taken into consideration the fact that the Android framework is split into three types of sensors: motion, environmental, and position.

The motion sensors include accelerometers, gyroscopes, rotational vectors, and gravity sensors. Their scope is to monitor the movement of the phone concerning the 3-axis (X, Y, and Z).

Environmental sensors are used to measure parameters such as ambient temperature, air pressure, humidity, or illumination. To be able to perform such tasks, the phone has barometers, thermometers, and photometers.

Position sensors refer to the actual position of the device, and location gathered also by orientation sensors and magnetometers.

Nowadays, it can be observed that Android-based devices have become more and more equipped with built-in sensors. These sensors are meant to monitor the environment where people live in order to improve their quality of life. For making this happen, sensors gather information about position, environment, or movement in the 3-axis system in raw format with high precision and accuracy.

The Android sensor framework can get raw output data from both categories of sensors: hardware and software.

In general, physical sensors measure directly specific environment properties such as geomagnetic field strength or acceleration as opposed to software-based sensors which only mimic the hardware behavior. The main characteristic of software-based sensors is the fact that they compute the data through specific algorithms and methods that take input from one or multiple hardware-based sensors. This type of sensor is sometimes called virtual or synthetic.

When monitoring the position of a device, the 3-axis coordinate system is taken into consideration, based on the screen orientation. Therefore, the X axis indicates the horizontal plane and points to the right side, the Y axis indicates the vertical plane and points to the upper side, and finally, the Z axis points toward the outside of the screen. According to this 3-axis system, anything behind the screen is in the negative values on the Z-axis. A few sensors which use this coordinate system are acceleration, gravity, gyroscope, and linear acceleration sensors.

The aggregation of data from the sensors enables the creation of usage patterns that could uniquely identify a device and the user of that device. By separating the device from the user, trust and reputation models related to wireless mobile networks could be improved. The device's unique identifier could be used to create a trust and reputation model for the hardware devices as proposed in (Mocanu et al., 2020) while the user profile could be used to improve a trust and reputation model like the one proposed in (Mihaita et al., 2017).

3.2 Communication Protocol

Furthermore, before the sensor data collection, the confidentiality of the communication channel must be handled. Therefore, a complex secure communication protocol is proposed based both on asymmetric cryptography and mobile device fingerprint.

In a simplified manner, Figure 3 illustrates the basic building blocks of the proposed communication protocol. It is assumed that each node in the spontaneous network has its pair of asymmetric keys and a verifiable, uniquely identifiable fingerprint. Also, the transmission environment should allow network broadcast messages.

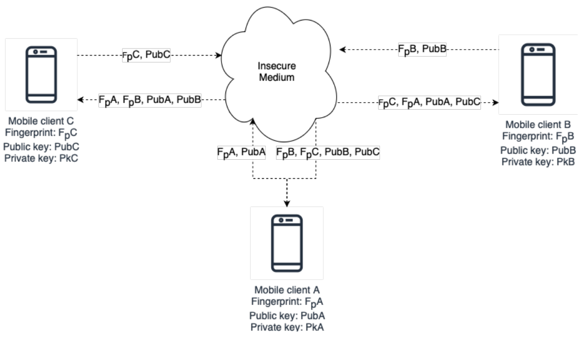

Figure 3. The simplified hand-shake phase of the communication protocol

When one node attempts to advertise its presence, it initiates a broadcast that contains its fingerprint and its public key, so each peer would learn the transmitted details. In this way, each receiving node would correlate the public key with the uniquely identifiable fingerprint. If another node broadcasts an identical pair of keys or fingerprints, it should be rejected by surrounding peers. In case the public key is used by an adversary to impersonate an existing node to transmit data, the decryption of communications would be unfeasible on the attacker's side.

After a node broadcasts its details, the surrounding peers, which are available for message passing, begin a consensus to choose the oldest two nodes to proceed towards the verification of the attending node. If the verified entity does not properly prove its identity, the verifier will propagate this information to prevent further connection attempts. If the node is validated, the censor peers advertise the result, to allow further interactions for the accepted participant.

To detect the nodes which are leaving, each node should broadcast a heartbeat message as described in (Hu et. al, 2023), concerning the channel bandwidth, at a regular time interval, *T_x*. After the heartbeat time passes, namely *DT_all*, all the participant nodes should mark the associated entry for deletion if the presence of a node was not confirmed. When the network becomes stable, all the participants are aware of the surrounding peers and their public keys and fingerprints. A detailed overview of the heartbeat broadcast message can be seen in Figure 4.

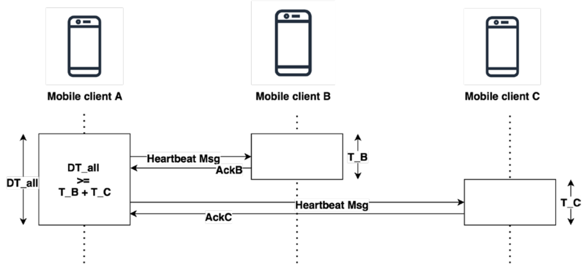

Figure 4. Heartbeat broadcast message protocol

3.3 Node Verification Protocol

Due to the spontaneous nature of the MANET, where new nodes appear and leave frequently, the new participant verification procedure, as shown in Figure 5, remains a big energy- and time-consuming task. Also, some decentralized networks consist of hundreds or even thousands of users, which means that a large quantity of data is processed and stored on mobile devices. In the present case, in order to avoid the overloading and the overheating of the hardware components, the maximum number of known peers of a given node is restricted to 1000 most frequently accessed participants.

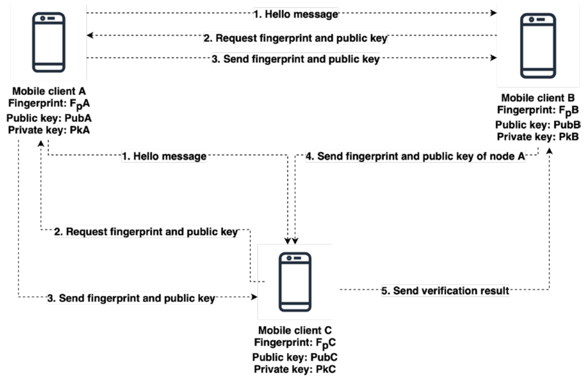

Figure 5. The verification scheme of a new participant

The new peer verification procedure starts when the "hello message" is broadcasted in the local area network (LAN), to announce its presence and to discover peers available for data dissemination. The principle behind this approach is like the DHCP-DISCOVER mechanism, relying on data exchange between the new node and the first peer which is available for message dissemination as shown in Figure 6.

After the "hello message" procedure is completed, node A sends its public key associated with its fingerprint to all the available peers, which will respond with their details. To maximize the trust between participants, the nodes available for data dissemination should verify the fingerprint of the new peer, but, currently, the solution proposed in this paper generates an internal value that could not be calculated from the outside of the physical device. Although a fingerprint could not be recalculated by others, the present solution verifies internally the presence of some of the most-known vulnerabilities, like root privileges, custom ROM installation, etc.

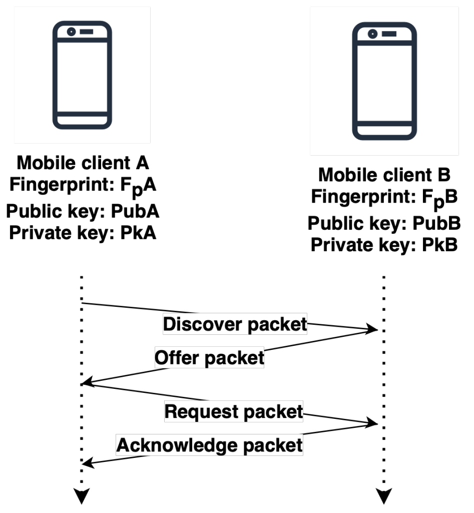

Figure 6. "Hello message" broadcasting mechanism

Also, the public key – fingerprint pair will be compared with the values known by the community members, to determine if the supplied information is unique and can be trusted for data dissemination purposes. This way, no malicious node could send a false message, impersonating a legit peer. If the verification procedure provides a negative result, the new node which tries to connect to the server is refused, and this information is disseminated between participants, to stop any message forwarding requests.

For a better understanding, the pseudocode of the verification scheme is presented in Algorithm 1.

Algorithm 1. Verification scheme

```
New node
Fp = get_device_fingerprint()
PubK = get_device_pubK()
PrivK = get_privateKey()
while (true)
        send_HelloPachet()
        recvFpRequest()
        send(Fp, PubK)
end while

Existing node
Fp = get_device_fingerprint()
PubK = get_device_pubK()
PrivK = get_privateKey()
while (true)
        recvFp(FpNew, PubNew)
        send(FpNew, PubNew, ExistingNode)
        recvVerificationResult()
end while
```

The process of node verification has a continuous nature since a peer can become malicious at any time, and without these security measurements, an attacker cannot be easily detected. When it comes to message dissemination procedures, the proposed solution relays on comparing public details of the targeted node with the data received from the nearby peers as shown in Figure 7, in order to overcome any action of impersonation. As it can be seen in Figure 7, if node D wants to send a message to B, it must verify the public key – fingerprint pair belonging to B with the values given to peers A and C. If the obtained result is negative, the sender will stop the data dissemination process and will notify all the other participants that node B is malicious, to stop any message forwarding requests.

If two devices decide to create a spontaneous network, the proposed solution cannot determine the degree of peer trust. In this case, if a new node tries to connect, the verification result cannot be trusted because the verifications have not been previously authorized. Also, the message dissemination mechanism can forward malicious data, because two nodes are insufficient for getting a consensual decision.

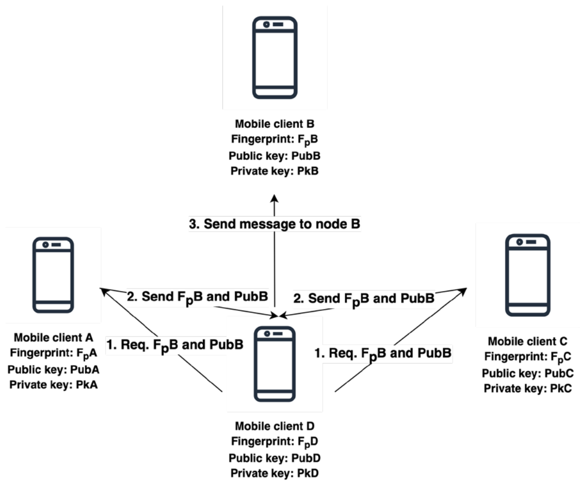

Figure 7. The verification scheme of an existing node

4. Use Case and Experimental Evaluation

The experiments for this paper are Android-oriented due to the large heterogeneity of the sensors and have been conducted on a Nexus 5 smartphone with Android operating system version 6.0.1 by two different users. The devices have the following sensors: compass, accelerometer, gyroscope, proximity, and barometer.

To generate a unique identifier for a node, it is necessary to find a set of spoofing-resistant verifiable hardware and software attributes. When it comes to mobile devices, there is some information that is hardly changeable, like the resolution of the screen, the hardware sensors list, the device manufacturer, and the operating system name. Some attributes can be obtained through non-elevated software API calls, and others require a high-security privilege context, which could pose a privacy and security risk (Shepherd et al., 2021). The proposed approach collects all data and then processes it to automatically compute and store the fingerprint value of the given physical device. The method used for generating the unique identifier is based on the SHA-256 algorithm, to obtain a 256-bit length fingerprint value.

The main target of the present research is to leverage the unprivileged API calls to reliably compute a fingerprint which could determine the uniqueness of a node because using highly privileged system calls is necessary to root/jailbreak the mobile device. The security risk associated with a highly privileged environment is given by the possibility for an attacker to execute instructions without the user's authorization, which is an unacceptable trade-off between security and usability.

Starting from this premise, the solution proposed in this research combines the information obtained from two categories of pseudo-unique identifiers: Non-Volatile Random-Access Memory (NVRAM) data and specific hardware features. The International Mobile Equipment Identity (IMEI), a value which is hardly changeable without root permissions, is extracted automatically from the NVRAM. The fingerprinting process is stopped if the solution identifies the existence of the *superSU* application or a custom firmware being installed.

4.1 Ad-Hoc Message Disseminator

The proposed solution took shape in the form of an Android application, named Ad-Hoc Message Disseminator, which currently has only a few features of the proposed solution like registering or logging in of a user and the fingerprinting module.

Starting from the premise that a user should not have to provide personal data about himself, he will choose only a username and a password for his registration. The provided password must meet the following conditions: a minimum of 10 characters, and the characters must be both alphanumeric and special. If conditions are not met, the registration procedure is resumed. To log in, the user provides only the selected credentials.

The fingerprinting module computes the uniquely identifiable value associated with a node and a public key. The first problem encountered was that emulated devices cannot provide a valid IMEI. For this reason, a proprietary function had to be implemented, in order to generate this value once per device.

From a security point of view, this approach does not require special permissions to increase the user's data confidentiality. To prevent any cyber-attacks on the user's integrity, the application verifies if the signature is identical to the one embedded in the source code. Also, it checks if the mobile device has privileged execution permissions to overcome any outside interventions.

To determine if the fingerprinting procedure can offer reliable results for uniquely identifying a specific participant of the Ad-Hoc network, we calculated the Shannon entropy of 100 calculated values. Due to the lack of a variety of physical mobile phones or tablets and usable GSM cards, the tests were performed on emulated devices, like Nexus versions 4, 5, 5X, 6, and 9 with Android operating system, versions 4.4.2 to 7.1.1.

Figure 8 presents the values obtained for hexadecimal fingerprint representation. The minimum and the maximum entropy values are 3.67 and 3.95, respectively, with an average value of 3.82. The scores are not so high, because only a small number of unique characters are used for hexadecimal representation.

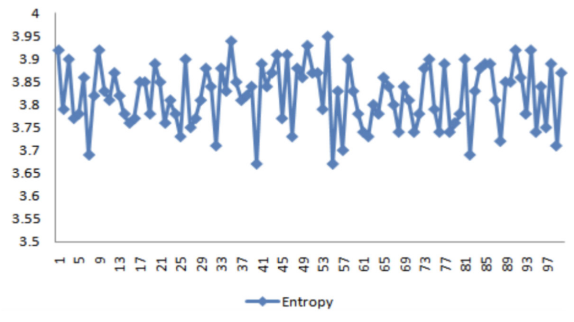

Figure 8. Shannon entropy analysis

To determine if the user who uses the mobile phone is the same as the one who owns the terminal, it is necessary to learn the user's behavioral patterns and calibrate the sensor's output data to a generic contextual environment and the associated motion states. Because gathering environmental and contextual data is related to the motion state of the device, as presented in (Kristoffersson & Lindén, 2022), the manner in which the sensor data is influenced by the user's motion status is firstly identified. Therefore, the implementation was tested using three different scenarios, as follows:

- when the user is in a stationary state (idle state);
- when the user is moving (walking state);
- when the user is driving or is in a car (driving state).

The present application was tested on a Nexus 5 mobile phone with Android operating system version 6.0.1. This device was chosen because the OS version is created and maintained by Google Inc., which does not have manufacturer applications or other undocumented software-based sensors.

Based on the three reference motion states, the raw output data was gathered over several days, in different contextual environments, and by different users to obtain as much information as possible. But, to correctly evaluate the obtained results, only the information gathered during the same day and generated by the same user was selected.

Figures 9, 10 and 11 show the AK8963 magnetometer sensor output data in three motion states: idle, walking, and driving. The oscillations which are shown by this sensor are determined by the sudden turns of the mobile device. The frequency of these oscillations rises mostly when the user is moving, because, although the car is moving, the stabilization sensor's component calibrates the raw data to prevent possible distortions caused by noise.

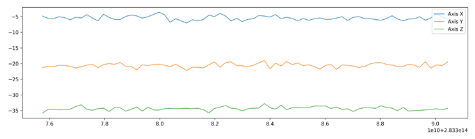

Figure 9. AK8963 magnetometer sensor output data when the user is in the idle state

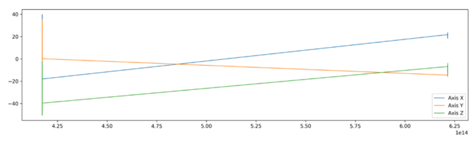

Figure 10. AK8963 magnetometer sensor output data when the user is walking

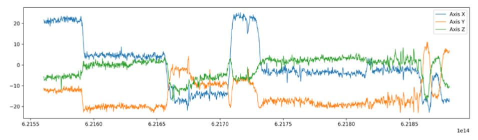

Figure 11. AK8963 magnetometer sensor output data when the user is driving

Figures 12, 13 and 14 show the APDS-9930 QPDS-T930-T930 proximity and light sensor output data. The level of light detected by this sensor varies based on the contextual time of day and the amount of sunlight. Also, this data may vary based on the user's behavioral patterns.

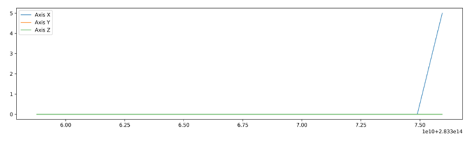

Figure 12. APDS-9930QPDS-T930 proximity & light sensor output data when the user is in the idle state

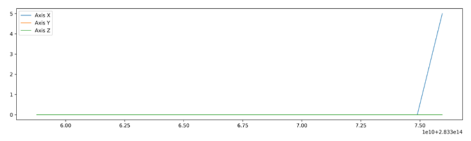

Figure 13. APDS-9930QPDS-T930 proximity & light sensor output data when the user is walking

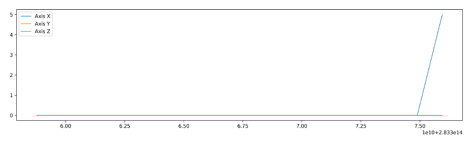

Figure 14. APDS-9930QPDS-T930 proximity & light sensor output data when the user is driving

Moreover, Figures 15, 16 and 17 present the BMP280 barometer sensor output data. Since this test was performed under constant weather and altitude conditions, the atmospheric pressure values remained constant throughout the whole experiment. However, fluctuations in sensor data might indicate that a user is gaining or losing altitude (e.g.: driving on a mountain road or flying with a plane).

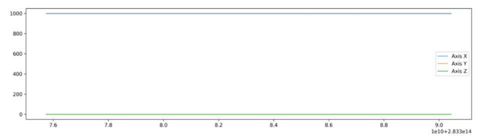

Figure 15. BMP280 barometer sensor output data when the user is in the idle state

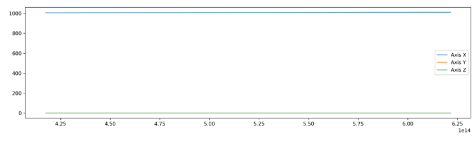

Figure 16. BMP280 barometer sensor output data when the user is walking

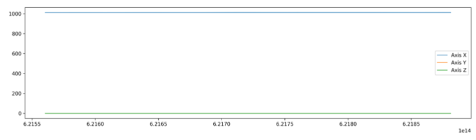

Figure 17. BMP280 barometer sensor output data when the user is driving

As it can be observed, both the proximity and light sensor and the barometer sensor tend to have similar output data. This fact indicates that these sensors are not affected by the user's motion state.

Furthermore, Figures 18, 19 and 20 present the output data from the GeoMagnetic rotation vector sensor. The GeoMagnetic rotation vector sensor acts like a compass, reading the user's moving direction. The spike in the stationary state was produced by the initial calibration procedure. The graph of motion depicts sudden changes in the direction of the mobile device, based on the initial point, while the graph of driving shows gradual oscillations caused by less frequent changes of the direction.

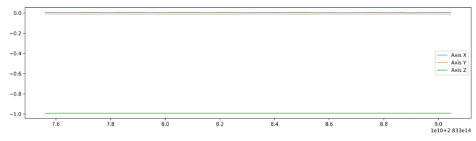

Figure 18. GeoMagnetic rotation vector sensor output data when the user is in the idle state

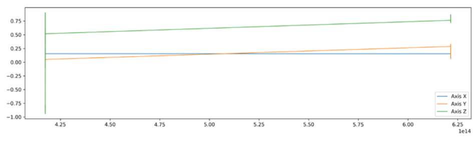

Figure 19. GeoMagnetic rotation vector sensor output data when the user is walking

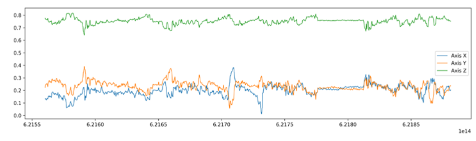

Figure 20. GeoMagnetic rotation vector sensor output data when the user is driving

Figures 21, 22 and 23 present the MPU6515 accelerometer sensor output data. As it can be observed, the accelerometer sensor detects sudden speed changes in the user's mobile device. In the graph of the driving state, it can be seen the fact that this sensor calibrates itself based on the previously measured values, to prevent abnormal maximum and minimum values.

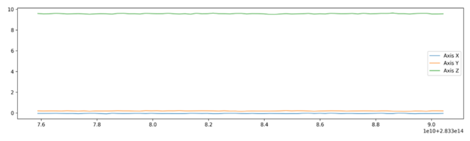

Figure 21. MPU6515 accelerometer sensor output data when the user is in the idle state

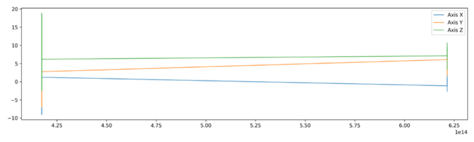

Figure 22. MPU6515 accelerometer sensor output data when the user is walking

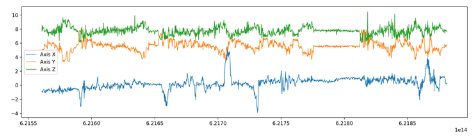

Figure 23. MPU6515 accelerometer sensor output data when the user is driving

Finally, Figures 24, 25 and 26 show the data associated with the MPU6515 gyroscope sensor. The gyroscope sensor is highly sensitive to external interaction factors. The graph of the stationary state illustrates the gyroscope output data affected by the laptop cooling fan.

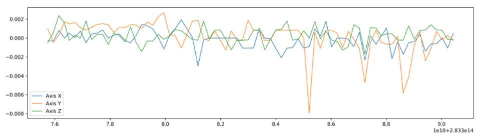

Figure 24. MPU6515 gyroscope sensor output data when the user is in the idle state

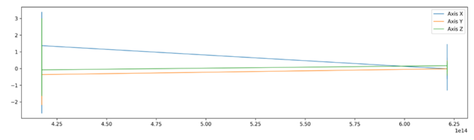

Figure 25. MPU6515 gyroscope sensor output data when the user is walking

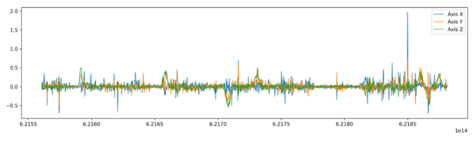

Figure 26. MPU6515 gyroscope sensor output data when the user is driving

5. Conclusion

The main objective of the present research focuses on the design of a multilayer architecture for distributed mobile systems that use device fingerprinting for behavioral identification, both for users and their devices. The proposed solution is a viable choice for distributed governance systems, especially for support decision systems, crowd management, or identity management.

This paper focuses on collecting and using mobile data sensors to identify the user's behavioral patterns, influenced by both the interaction with the environment and by human-machine interaction, as perceived by an Android-based sensor stack. The obtained results have shown the fact that modern mobile smartphones can sense the presence of a living being they are interacting with and can provide some information about the usage context.

This approach might provide meaningful insights about user behavioral patterns without any special permission and without disclosing the real owner's identity or biometric data. Sensor data could be aggregated with built-in biometric authentication systems, to verify and confirm the ownership of the device. This approach can also be used to enhance the biometric authentication process, to prevent biometric spoofing concerning data protection regulations (GDPR).

Another important aspect of the present work is represented by the differentiation between the hardware device and its user, with applicability in real-time trust and reputation models for distributed mobile networks, especially in governance systems.

Similar sensor data aggregation is implemented by various Android-based terminal vendors, by using proprietary hardware. However, this technology is a matter of speculation from a security standpoint because it might be reused to leak sensitive information about the user's context.

Future work regarding the hybrid processing layer presented in section 3 is planned to be investigated. The objective is to increase both the number of devices on which the application runs and the number of users, in order to establish, with a high degree of accuracy, the probability to which the identity of the users remains the same across devices. Also, multiple components will be included in the processing layer such as learning engines based on various SVM, and neural network kernels, a security component for anonymization and confidentiality of the processed data, and a real-time scheduler for cloud processing optimization. On the edge side of the processing layer, basic sensor data tasks will be researched and developed, such as data normalization, outlier detection, and data fusion.

Acknowledgments

The research presented in this paper has been funded by the Operational Programme Human Capital, POCU/993/6/13/153735 – OPTIM Research. This work has been supported by the grant POCU/993/6/13/153178, co-financed by the European Social Fund within the Sectorial Operational Program Human Capital 2014 – 2020. Also, this paper has been supported by project FARGO: Federated leARninG for human moBility (PN-III-P4-ID-PCE-2020-2204), the project CloudPrecis (SMIS code 2014+ 124812) and the project ODIN112 (PN-III-P2-2.1-SOL-2021-2-0223).

REFERENCES

Alaca, F. & van Oorschot, P. C. (2016). Device fingerprinting for augmenting web authentication. In: *Proceedings of the 32nd Annual Conference on Computer Security Applications, 05 – 09 December 2016, Los Angeles, CA, USA*. New York, USA, Association for Computing Machinery (ACM). pp. 289–301. doi: 10.1145/2991079.2991091.

Al-Shareeda, M. A. & Manickam, S. (2022) Man-in-the-Middle Attacks in Mobile Ad Hoc Networks (MANETs): Analysis and Evaluation. *Symmetry*. 14(8), 1543. doi: 10.3390/sym14081543.

Benson, K., Dainotti, A., Claffy, Kc C., Snoeren, A. C. & Kallitsis, M. (2015) Leveraging Internet Background Radiation for Opportunistic Network Analysis. In: *Proceedings of the 2015 Internet Measurement Conference, 28 – 30 October 2015, Tokyo, Japan*. New York, USA, Association for Computing Machinery (ACM). pp. 423-436. doi: 10.1145/2815675.2815702.

Ciobanu, R.-I., Marin, R.-C., Dobre, C. & Cristea, V. (2017) Trust and reputation management for opportunistic dissemination. *Pervasive and Mobile Computing*. 36, 44–56. doi: 10.1016/j.pmcj.2016.09.016.

Darsena, D., Gelli, G., Iudice, I. & Verde, F. (2022) Sensing Technologies for Crowd Management, Adaptation, and Information Dissemination in Public Transportation Systems: A Review. *arXiv.* [Preprint] https://arxiv.org/abs/2009.12619. [Accessed 16th May 2023]. doi: 10.36227/techrxiv.16811191.

Fukumoto, N., Ano, S. & Goto, S. (2013) Passive Smart Phone Identification and Tracking with Application Set Fingerprints. In: *Proceedings of the Asia-Pacific Advanced Network*. 36, pp. 41-48. doi: 10.7125/apan.36.6.

Hu, Y., Tian, G., Jiang, A., Liu, S., Wei, J., Wang, J., & Tan, S. (2023) A Practical Heartbeat-based Defense Scheme Against Cloning Attacks in PoA Blockchain. *Computer Standards & Interfaces*. 83, 103656. doi: 10.1016/j.csi.2022.103656.

Hupperich, T., Maiorca, D., Kührer, M., Holz, T. & Giacinto, G. (2015) On the Robustness of Mobile Device Fingerprinting. In: *Proceedings of the 31st Annual Computer Security Applications Conference, 07 – 11 December 2015, Los Angeles, CA, USA*. New York, USA, Association for Computing Machinery (ACM). pp. 191–200. doi: 10.1145/2818000.2818032.

Jayachandran, S. & Jaekel, A. (2022) Adaptive Data Rate Based Congestion Control in Vehicular Ad Hoc Networks (VANET). In: Bao, W., Yuan, X., Gao, L., Luan, T. H. & Choi, D. B. J. (eds.) *Ad Hoc Networks and Tools for IT. ADHOCNETS 2021, TridentCom 2021. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*. 428, Cham, Switzerland, Springer International Publishing, pp. 144–157. doi: 10.1007/978-3-030-98005-4_11.

Katila, R., Gia, T. N. & Westerlund, T. (2022) Analysis of mobility support approaches for edge-based IoT systems using high data rate Bluetooth Low Energy 5. *Computer Networks*. 209, 108925. doi: 10.1016/j.comnet.2022.108925.

Korir, F. C. & Cheruiyot, W. (2022) A survey on security challenges in the current MANET routing protocols. *Global Journal of Engineering and Technology Advances*. 12(1), 78–91. doi: 10.30574/gjeta.2022.12.1.0114.

Kristoffersson, A. & Lindén, M. (2022) A Systematic Review of Wearable Sensors for Monitoring Physical Activity. *Sensors*. 22(2), 573. doi: 10.3390/s22020573.

Kurtz, A., Gascon, H., Becker, T., Rieck, K. & Freiling, F. (2016) Fingerprinting Mobile Devices Using Personalized Configurations. In: *Proceedings on Privacy Enhancing Technologies*. 2016(1), pp. 4–19. doi: 10.1515/popets-2015-0027.

Li, J., Dai, J., Issakhov, A., Almojil, S. F. & Souri, A. (2021) Towards decision support systems for energy management in the smart industry and Internet of Things. *Computers & Industrial Engineering*. 161, 107671. doi: 10.1016/j.cie.2021.107671.

Majumder, S. & Deen, M. J. (2019) Smartphone Sensors for Health Monitoring and Diagnosis. *Sensors*. 19(9), 2164. doi: 10.3390/s19092164.

Mihaita, A., Dobre, C., Pop, F., Mocanu, B., Cristea, V. & Esposito, C. (2017) A Trust Application in Participatory Sensing: Elder Reintegration. In: Au, M., Castiglione, A., Choo, K. K., Palmieri, F. & Li, K. C. (eds.) *Green, Pervasive, and Cloud Computing. Lecture Notes in Computer Science*. Cham, Switzerland, Springer International Publishing, pp. 596–610. doi: 10.1007/978-3-319-57186-7_43.

Mocanu, A. E., Mocanu, B. C., Esposito, C. & Pop, F. (2020) Trust is in the air: A new adaptive method to evaluate Mobile wireless networks. In: *Testing Software and Systems: Proceedings of the 32nd IFIP WG 6.1 International Conference (ICTSS 2020), 09 – 11 December 2020, Naples, Italy*. Verlag, Berlin, Heidelberg, Springer International Publishing. pp. 135-149. doi: 10.1007/978-3-030-64881-7_9.

Sánchez Sánchez, P. M., Jorquera Valero, J. M., Huertas Celdrán, A., Bovet, G., Gil Pérez, M. & Pérez, G. M. (2023) A methodology to identify identical single-board computers based on hardware behavior fingerprinting. *Journal of Network and Computer Applications*. 212, 103579. doi: 10.1016/j.jnca.2022.103579.

Shepherd, C., Kalbantner, J., Semal, B. & Markantonakis, K. (2021) A side-channel analysis of sensor multiplexing for covert channels and application fingerprinting on mobile devices. *arXiv.* [Preprint] https://arxiv.org/abs/2110.06363. [Accessed 16th May 2023]. doi: 10.48550/arXiv.2110.06363.

Sicari, S., Rizzardi, A. & Coen-Porisini, A. (2022) Insights into security and privacy towards fog computing evolution. *Computers & Security*. 120, 102822. doi: 10.1016/j.cose.2022.102822.

Wei, D., Gu, Y., & Du, Y. (2022) Mobile Device Fingerprinting Recognition using Insensitive Information. In: *2022 International Conference on Image Processing, Computer Vision and Machine Learning (ICICML), 28 – 30 October 2022, Xi'an, China*. USA, IEEE. pp. 1-6. doi: 10.1109/icicml57342.2022.10009697.

Zukarnain, Z. A., Muneer, A. & Ab Aziz, M. K. (2022) Authentication Securing Methods for Mobile Identity: Issues, Solutions and Challenges. *Symmetry*. 14(4), 821. doi: 10.3390/sym14040821.