Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/cose

Computers & Security

An approach to detect user behaviour anomalies within identity federations

Alejandro G. Martín, Marta Beltrán, Alberto Fernández-Isabel, Isaac Martín de Diego*

Rey Juan Carlos University, Department of Computing, ETSII, C/ Tulipán, s/n, Móstoles, 28933, Madrid, Spain

ARTICLE INFO

Article history:
Received 13 January 2021
Revised 22 April 2021
Accepted 31 May 2021
Available online 10 June 2021

Keywords:
Anomaly detection
Behavioural fingerprint
Federated identity management
Machine learning
User and entity behaviour analytics

ABSTRACT

User and Entity Behaviour Analytics (UEBA) mechanisms rely on statistical techniques and Machine Learning to determine when a significant deviation from patterns or trends established as a standard for users and entities is occurring. These mechanisms are beneficial within cybersecurity contexts because they allow managers and administrators to have early alerts warning about potential security incidents. This paper proposes the utilisation of UEBA to improve the security of Federated Identity Management (FIM) solutions. The proposed UEBA workflow allows Relying Parties within identity federations to build a session fingerprint characterising each user's behaviour from available information. Furthermore, it enables anomaly detection based on this fingerprint, integrating raised alerts within current identity management specifications. The proposed workflow is validated and evaluated in a real use case based on a web chat application using OpenID Connect for identity management.

1. Introduction

User and Entity Behaviour Analytics (UEBA) relies on Machine Learning (ML) to model users and entities' behaviour trying to find anomalous behaviour that could be the sign of a cyberattack. UEBA solutions usually gather information on the average expected behaviour of users and entities from different sources. Once this information is filtered and pre-processed, a baseline of user behaviour can be established through patterns or fingerprints. Then, UEBA solutions perform continuous monitoring of users and entities' behaviour to compare it to baseline behaviour.

This work focuses on proposing a framework to add UEBA techniques to Federated Identity Management (FIM) solutions (Chadwick, 2009) such as OpenID Connect or Mobile Connect OIDF (2021). With these identity management specifications, end-users credentials are stored at an external server or Identity Provider (IdP), responsible for Identification, Authentication, Authorisation and Accounting (IAAA). When an end-user needs to access a resource, application or service (i.e. the Relying Party or RP), the RP trusts the external server or IdP to solve IAAA. Thus, the end-user is authenticated outside the RP (i.e., in the IdP), obtaining a capacity in the form of a token. Finally, the end-user can access the RP using this token. Moving from traditional solutions to identity federations implies that the authentication process goes from local (i.e., the RP stores and checks locally the end-user credentials or authenticators) to outsourced (i.e., the RP trusts the IdP in order to accomplish the IAAA).

* Corresponding author.
E-mail addresses: alejandro.garciam@urjc.es (A.G. Martín), marta.beltran@urjc.es (M. Beltrán), alberto.fernandez.isabel@urjc.es (A. Fernández-Isabel), isaac.martin@urjc.es (I. Martín de Diego).
https://doi.org/10.1016/j.cose.2021.102356

All mentioned specifications are mature standards mainly used within network and web environments. However, all of them have security vulnerabilities due to their specification, or bad implementation (Navas and Beltrán, 2019). A potential solution to enhance their security is to use UEBA techniques at the RP. It is often assumed that when an RP delegates the IAAA to an IdP, working with an FIM solution, the RP does not have to worry about security, transferring all the security burden to the IdP. Nevertheless, outsourcing the authentication function does not mean outsourcing security from the prevention or detection point of view. The RP might have a significant role when protecting users from spoofing or impersonation attacks, given that users interact with the RP almost all the time. Relying on an FIM solution does not imply that security is only an IdP responsibility. Understanding that both the IdP and the RP have their role in securing the IAAA flows, the end-user will be more protected from identity federations' vulnerabilities. To the best of our knowledge, the combination of both areas, UEBA and FIM, has not been addressed. Despite the exciting synergies that can emerge between the two.

Take the following as an example of the great benefits of these synergies. Imagine that an end-user authenticates on a flight ticket purchase platform relying on an IdP (Facebook, Google, etc.). To perform authentication, the end-user enters her credentials in the browser so that they can be sent to the IdP and validated. Once these credentials are validated, the IdP provides the user with the ID Token and Access Token to access the resources of the ticket purchasing platform. If an attacker gains access to the victim's laptop, these tokens can be stolen from the browser cache, for example. This allows the attacker to take control and have full access to the flight ticket purchase platform on behalf of the victim, as long as the tokens remain valid. In case the RP has implemented a security mechanism based on UEBA, it could detect that the attacker's behaviour is anomalous, as it does not resemble the victim's behaviour when navigating the web or buying tickets. In this example, the RP would send a token revocation request to the IdP, kicking the attacker out of the system and requesting the credentials again. Note that the attacker could repeat the process and regain access. This paper proposes different approaches to solve different use cases, such as requiring a higher level of assurance of the end-user's identity. An attacker could not repeat this attack as this requirement could involve including a second authentication factor. Alternatively (or, in addition), the RP may warn the end-user of potential security breaches to trigger corrective actions (change the password, restrict physical access to her laptop, etc.).

This work's main contribution is to propose and validate a novel UEBA workflow that can be followed at RPs to improve the security levels currently provided by IdPs, enabling better prevention and detection of impersonation attacks based on credential theft or session hijacking. Thus, the guidelines to specify a session fingerprint that summarises users' behaviour when interacting with a resource, service or application (i.e., the RP) are provided. Moreover, a novel approach for detecting behaviour anomalies based on this fingerprint using ML techniques is detailed. Finally, the proposed approach is validated in a real scenario, testing its functionalities and assessing its efficiency and security levels.

The rest of this paper is organised as follows. Section 2 presents an overview of the related work and the main motivations to perform this research. Section 3 introduces the considered architecture and assumptions. Section 4 describes the proposed approach to include a UEBA workflow at any RP within an identity federation. Section 5 details the proposed solution implementation and validation with a real chat application. Finally, Section 6 summarises our main conclusions and the most interesting lines for future research.

2. Related work and motivation

2.1. *User and entity behaviour analytics to improve security*

UEBA is focused on modelling behaviours to improve confidentiality, integrity or availability within cybersecurity contexts. It is usually addressed like a classification problem. In particular, it is tackled as an outlier detection problem (using ML terminology) (Hodge and Austin, 2004) or like an anomaly detection problem (using cybersecurity terminology) (Chandola et al., 2009). Different research works have addressed this problem in the last years. Table 1 shows some of the most significant to this research.

All these works can be firstly categorised according to the analysed subject, a user or an entity. The first group of works, proposing new ways of analysing users' behaviour, can be categorised again depending on the used device, which is the most common means to obtain up-to-date and accurate human behaviour features. These devices are, typically, personal computers or smartphones, but any device from which behavioural metrics could be extracted might be used (e.g. smartwatches or tablets). Keystroke or mouse dynamics are usually chosen to model the behaviour when using personal computers, while sensors information (e.g. touchscreen, accelerometer and gyroscope) are selected when using portable devices.

Moreover, devices considered in previous works are usually connected to a network. The characteristics of the sent or received packages and information regarding network protocols at different layers or metadata can help when modelling users' behaviour. This can be observed in works focused on identifying users while surfing the Internet, which have as their primary objectives personalised advertising or security.

This first group analysing users relies on some kind of behavioural fingerprint, a detailed, nearly unique, difficult to fake, and durable over time identifier or marker. This is typically the case of researches focused on intrusion detection, attack attribution and forensics, or other application domains requiring the comparison of users or normal/abnormal behaviours. On the other hand, other works focusing on authentication or continuous authentication do not need to build this identifier.

The second group of previous works analyse the behaviour of entities, and its size is growing due to the emergence of the Internet of Things (IoT). Again, there are different groups in this category attending to the considered device. In this case, a sensor, an actuator, a controller and a hub. Again, different

Table 1 – Comparison of previous works.

Work	User/ Entity	Device	Behavioural features	Fingerprint	Implementation	Purpose
Ikuesan and Venter (2019)	User	PC	Mouse	Yes	Specific	Forensic/ Identification
Voris et al. (2019)	User	PC	File system actions/ Network traffic	No	Specific	Authentication
Meng et al. (2018b)	User	Smartphone	Touchscreen	No	Specific	Authentication
Chow et al. (2010)	User	Smartphone	Smartphone sensors/ network traffic	No	Framework	Authentication
Meng et al., 2020	User	Smartphone	Smartphone usage	No	Specific	Insiders detection
Smith-Creasey and Rajarajan (2019)	User	Smartphone	Smartphone sensors/ Gestures	No	Specific	Authentication
Killourhy and Maxion (2009)	User	PC	Keystrokes	No	Specific	Authentication
Ahmed et al. (2008)	User	PC	Keystroke	Yes	Specific	Intrusion detection
Gascon et al. (2014)	User	Smartphone	Touchscreen	Yes	Specific	Authentication
Shen et al. (2017)	User	Smartphone	Smartphone sensors	No	Specific	Authentication
Xiaofeng et al. (2019)	User	PC	Keystroke	No	Specific	Authentication
Bhana and Flowerday, 2020	User	PC	Keystroke	No	Specific	Authentication
de Fuentes et al. (2018)	User	Smartphone	Non-Assisted smartphone Sensors	No	Specific	Authentication
Mondal and Bours (2016)	User	PC	Keystroke/ Mouse	No	Specific	Authentication
Herrmann et al. (2014)	User	ALL	Network traffic	Yes	Specific	Forensic/ Identification
Lackner et al. (2010)	User	ALL	Email usage	Yes	Specific	Enhancing privacy/ Threat detection
Bakar and Haron (2014)	User	ALL	Browser, Geolocation	No	Centralised Framework	Authentication
Taneja (2013)	Entity	Sensors	Mobility indicators	No	Framework/ M2M protocols	Compromised devices
Gu et al., 2018	User/ Entity	ALL	Network traffic/ Runtime Environment	Yes	Specific	Identification
Bezawada et al. (2018)	Entity	Sensors	Network traffic/ Response sequences	Yes	Specific	Entity identification
Formby et al. (2016)	Entity	ALL	Network traffic	Yes	Specific	Identification
Huda et al. (2019)	Entity	ALL	Malware data/ Runtime Environment	No	Specific	Malware detection
Sato et al. (2016)	Entity	Sensors	-	No	Framework	Establish trust
Shahid et al. (2018)	Entity	Sensors	Network traffic	No	Specific	Identification
Thangavelu et al. (2018)	Entity	Sensors	Network traffic	No	Specific	Identification
Yang et al. (2019)	Entity	Smartphone	side-channel information	yes	Specific	side-channel attacks

(*continued on next page*)

Table 1 (*continued*)

Work	User/ Entity	Device	Behavioural features	Fingerprint	Implementation	Purpose
Li et al., 2020	User/Entity	IoT	Communications	No	Specific	Improving Communication
This work	User/Entity	ALL	Free to choose	Yes	FIM standards	IAAA

behaviour features can be considered, although in this case, they are always related to network traffic, protocols, load, etc. Some works rely on a fingerprint characterising each device's behaviour, while others do not need this kind of identifier.

A significant difference with the first group of works is that resource constraints at the devices prevent, in many cases, the execution of UEBA techniques on the own device, requiring to offload processing and computation to a full-resource external server (something that is not usual in works focused on UEBA)

However, the two groups, those who analyse users and those who analyse entities, have more aspects in common. Firstly, the application domains already mentioned, such as identification, authentication or intrusion detection. Secondly, the revisited works (see Table 1) are majorly based on specific technological stacks and under quite significant assumption sets. In other words, they are not very portable or reproducible in different contexts for which they were initially proposed.

2.2. *Motivation and use cases*

There are quite a lot of use cases that could benefit from the approach proposed in this research. In general, all RPs requiring an improvement of the security levels provided to their end-users through new anti-spoofing capabilities such as risk-based identity assurance levels, continuous authentication or early alert systems. Good examples can be found in e-commerce, finance and banking applications and services using Personal Identifiable Information (PII). In this paper, the following use cases have been considered as representative of the context being researched:

- Use case 1: An RP decides to use the Level of Assurance concept in its requests to the IdP (something that is already specified in Mobile Connect but not in the rest of FIM specifications such as SAML Oasis, 2021, OAuth IETF, 2021, or OpenIDConnect OIDF, 2021). Therefore, the RP can specify in its request the degree of confidence required for the end-user authentication at the IdP (one or two authenticators, biometry, cryptography or in-person identity proofing). The RP would like to set this Level of Assurance (LoA) dynamically, taking into account the risk involved with each specific request. For example, if this RP is an e-commerce site, the LoA could be raised because the end-user is trying to make a purchase from a different browser and with a completely different mouse usage dynamic than in the previous sessions. Something that is considered suspicious, of course. It could have a reasonable explanation, but it could also be malicious spoofing due to an IdP credentials compromise.
- Use case 2: An RP decides to perform continuous authentication during the end-user session, for example, because it is an app dealing with the medical history of the end-user. Once the IdP has authenticated the end-user, the RP may force the IdP to perform a new authentication and to refresh the token if anomalous behaviour is observed for a specified period. Again, it could have a reasonable explanation, but it could also be a session hijacking.
- Use case 3: An RP, for example, regarding personal finance management, decides to store behavioural information gathered during users' sessions. This information is not processed in real-time as in the two previous use cases, but it can be processed periodically (every night, every weekend) in order to detect potential security breaches in retrospect and to ask the end-user to take the necessary precautions (e.g. changing the password at the IdP or the bank).

These three use cases also apply in scenarios where federated management is used to authenticate entities (sensors, robots, IoT devices) instead of people. These scenarios are increasingly used (Beltrán, 2018; Sciancalepore et al., 2017) in critical application domains where the information shared between the entity and the RP might be related to the location of people, intellectual property or medical data.

The limitations identified in previous works when focusing on these use cases can be summarised in:

- Previous works that propose using the LoA concept to improve security levels of identification or authentication processes specify the LoA requested value statically. They cannot assess users or entities' behaviour to decide the best value at a given moment considering the usability-cost-security trade-off.
- Previous works that propose using UEBA to solve authentication, access control or IAAA are always focused on adaptive authentication, distributed or centralised, but they are not suitable for federated approaches involving an IdP, an external or third-party provider.
- Previous works using UEBA to detect cyberattacks or fraud are tied to very particular application domains. These approaches have not proposed generic models or techniques reusable at very different RPs. Furthermore, when relying on different FIM specifications.

This research aims to overcome all these limitations by integrating UEBA techniques at the RP with the definition of a

new workflow. More specifically, the proposed workflow begins with defining a session fingerprint at the RP, summarising the most significant attributes of users' behaviour. Then, we propose anomaly detection techniques based on this fingerprint that might be used in the introduced use cases: pre or post-authentication (use case 1, 2 and 3 respectively) and online or offline (use cases 1, 2 and 3 respectively), at different RPs with different security requirements and computing capabilities. No previous works allow us to solve these issues, even less when an easy integration with current FIM specifications is required.

3. Architecture and assumptions

In this research, the three-role architecture of identity federations is considered: the end-user (EU), the Identity Provider (IdP) and the Relying Party (RP).

RPs (i.e. resources, applications, and services accessed by end-users) require knowledge about these users to optimise resources, improve their experience, or personalise offered contents, to mention only some examples. This knowledge is obtained from different technologies, interfaces and APIs capable of retrieving detailed information regarding the hardware and software configuration from browsers and mobile apps and user behaviour. Therefore, it can be used to build a unique identifier called in this research a session fingerprint.

Session fingerprinting might raise privacy concerns because it can perform profiling or be used to deliver personalised advertising. Different researchers and private companies have developed these last years countermeasures to protect browsers and apps from fingerprinting techniques. Almost all these techniques focus on script blocking or changing the values of gathered fingerprint's attributes (Vastel et al., 2018a).

In this paper, it is assumed that:

1. There is enough users behaviour diversity to build unique session fingerprints at the RP (Eckersley, 2010; Laperdrix et al., 2016).
2. The RP does not need the collaboration of the IdP to build this fingerprint nor to analyse it for anomalies; it can follow the proposed workflow with its own resources.
3. The RP informs the end-user that this type of techniques will be used to improve the levels of security it offers, in particular, to avoid impersonation due to credential theft or session hijacking.
4. The RP complies with the privacy and data protection regulations (such as GDPR). It also complies with the privacy-by-design principles, minimising information collection, obtaining the necessary explicit and informed consent from the end-user, and guaranteeing the appropriate levels of transparency.

4. Our approach: UEBA workflow

The proposed workflow provides some general guidelines for each RP to follow to reach their specific solution depending on the application domain and the specific use case. Remember that some of these use cases are pre-authentication and others post-authentication, some are online and others offline, available computing resources or consent collected from the users at RPs may be diverse, etc.

In Fig. 1, the whole workflow is illustrated. Note that the steps shown are not enumerated. That is because these steps may be asynchronous. Besides, the tasks that need to be accomplished explicitly by the RP to include UEBA are shown in Fig. 2. It has to be highlighted that the process may iterate backwards and forwards depending on the partial results obtained in each task.

First, the session fingerprint has to be specified. Relevant and descriptive attributes have to be selected and included in the fingerprint to enable the anomaly detection process. The second task is focused on gathering data from end-users, storing it efficiently and generating fingerprints. The technology and procedures to obtain and store the required information have to be designed and deployed. The data have to be filtered, merged, pre-processed and transformed, so a robust and descriptive session fingerprint is available for each end-user. A behavioural fingerprint will probably change over time. Thus, procedures to verify if a fingerprint is still valid or to update it when required must be defined.

The third task is modelling fingerprints. Specific ML techniques have to be considered, specifically anomaly detection techniques. These techniques allow finding events that do not fit with the baseline behaviour. In our particular case, abnormal events correspond to user behaviours caused by a security breach. These behaviours are extracted from the fingerprint generated in the previous task. The primary considerations an RP should consider to achieve good results are selecting a functional similarity or distance function that allows a fair comparison between fingerprints. Then, setting a descriptive threshold that enables detecting the oddest behaviours. Finally, defining the corresponding procedures to update the models to consider the latest behaviours that arrive at the system. For this one, an example is re-training the model over time or selecting an online model that trains at the same time it makes predictions.

The fourth task is the evaluation of the modelling task results. This process enables detecting essential features missing in the fingerprint or possible weaknesses in the selected anomaly detection techniques. The detected weaknesses can help in improving the fingerprint (going back to the first task), the process of its generation (going back to the second task) or the selected models (going back to the third task). Standard evaluation metrics (e.g. accuracy) are not descriptive enough due to the nature of the problem. Thus, specific metrics have to be considered and analysed to determine if the selected techniques are suitable for each specific use case.

Finally, the fifth task comprises the integration within an FIM solution. This process depends on the specific identity federation and the related use case. The most relevant considerations that should be made during the integration task are minimising the required modifications to the standard IAAA flows, using the mechanisms provided by those standards to modify them if necessary and informing users that new data will be collected and processed.

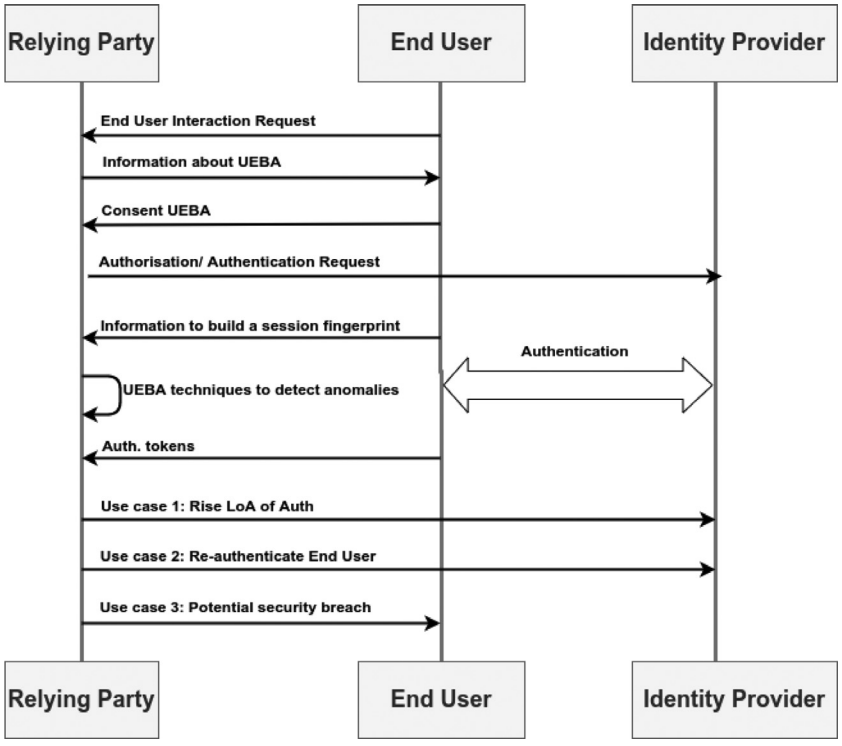

Fig. 1 – UEBA workflow.

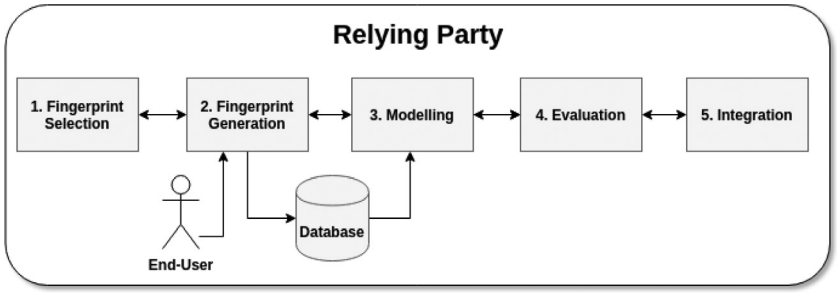

Fig. 2 – UEBA workflow at the Relying Party.

4.1. *Fingerprint selection*

This work proposes a set of attributes that any RP could use to build, from current and accessible technologies, a useful session fingerprint to complete the proposed workflow's first task. Each RP will have to decide the minimum set of attributes that guarantees the necessary security levels. It is not possible to define one specific fingerprint because it is not a one-solution-fits-all issue. There are RPs of a very different nature, with very different interaction with end-users and very different available resources within very heterogeneous use cases and application domains.

It has to be highlighted that there is an efficiency-privacy trade-off in the session fingerprint attributes selection: the more attributes that are considered, the easier it will be to differentiate one user from another, and the more efficient UEBA will be in detecting anomalies. On the other hand, the more expensive it will be to collect, process and store these fingerprints and the more invasive the RP will be with the user's privacy. Again, protecting sensitive personal information or a financial transaction is not the same as protecting the checking in for a flight. Compliance with privacy and data protection regulations will be a factor in adopting our workflow.

All proposed attributes, summarised in Fig. 3, can be categorised as static or dynamic. Static attributes are composed, mainly, by context information. Thus, these attributes tend to change little or none over time. On the other hand, dynamic attributes represent how an end-user interacts with his or her environment. These attributes can be significantly variable. Note that there is a defined set of possible values for static attributes, so there may be more overlap between different end-users values, whereas dynamic attributes may take any value (e.g., there are no unique behavioural patterns that group most users together). The group of static attributes guaranteeing sufficient entropy or diversity that could be included in the session fingerprint is the following, being our recommenda-

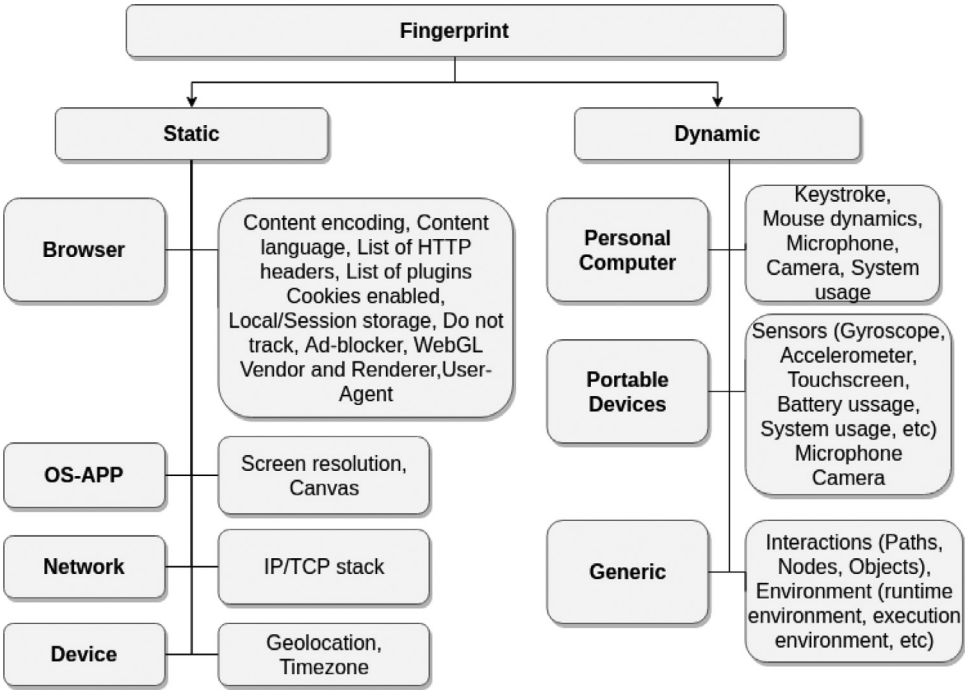

Fig. 3 – Summary of possible fingerprint attributes.

tion to select a combination of browser-based features, operating system and application features, network features and hardware features:

- Content encoding: Which content-encoding, usually a compression algorithm, the browser is able to understand.
- Content language: The user's own preferred language when using the browser.
- List of HTTP headers: String containing operating parameters for future HTTP transactions.
- List of plugins: Plugins installed within the browser context. For example, Java, Flash, or Silverlight.
- Cookies enabled: Configuration regarding cookies at the browser.
- Use of local/session storage: Supported browser storage mechanisms (localStorage, indexedDB, sessionStorage, Web-SQL via openDatabase, etc.).
- Do not track: Users add this header when they want to opt-out of tracking performed by some websites.
- Use of an ad-blocker: If the end-user has installed software to block aggressive or privacy-invasive advertising at the browser.
- WebGL Vendor and Renderer: WebGL is an API for rendering graphics within browsers. This API can expose different attributes of the underlying browser.
- User-agent: A string including the browser/operating system vendor and version, system language and platform.
- IP address: To identify the communication source,
- TCP/IP stack: Features such as open ports, DNS resolver, being behind a NAT, etc.
- Geolocation: Obtained with network-based mechanisms or via specific APIs.
- Screen resolution, colour depth and pixel density: Depending on the graphics card and the installed driver.
- Timezone: Device's time zone, observance of daylight saving time, or clock drift from Coordinated Universal Time (UTC).
- Canvas: Scripting can be used to render text and graphics in an HTML canvas and send back to the RP a hash of the resulting bitmap. Different devices generate different images with different software or hardware configurations (fonts, font rendering, emojis).

Regarding dynamic attributes, they can be strongly dependent on the device used to access the RP. For example, keystroke, mouse, microphone or camera dynamics are helpful when accessing from personal computers or laptops. On the other hand, behavioural dynamics can be extracted from sensors (e.g. gyroscope, accelerometer, and touchscreen) from portable devices like smartphones or tablets. The usage of resources such as the battery or the microphone/camera could be included in the fingerprint too. Some behavioural dynamics can be used with all devices, for example, those regarding user interactions with the RP's interface as the specific paths, nodes or objects used to accomplish a task, or the transaction rate. The environment in which an interaction is being performed may also be considered (e.g. runtime environment or execution environment).

In this work, RP profiles have been defined to help RPs to define the first fingerprint. The attributes recommended to each profile are based on previous state-of-the-art studies that analyse the uniqueness of available attributes. Other factors that have been considered are the collectability of each attribute in the considered scenario, the potential acceptability of end-users (that they do not perceive it as an invasion

of their privacy), the compliance with regulation, and the resource consumption that collecting/processing an attribute may suppose. For the static attributes, it is worth mentioning the studies (Gómez-Boix et al., 2018; Laperdrix et al., 2020), while for the dynamic attributes (Abuhamad et al., 2020; Bhatnagar et al., 2013). Web and mobile RPs have been distinguished while three target security levels have been considered (basic, medium and high):

- Basic security, web RP: User-agent, list of plugins, list of fonts, screen resolution, timezone and cookies enabled.
- Basic security, mobile RP: Screen resolution, timezone, battery usage, system usage.
- Medium security, web RP: Basic web, Canvas, WebGL, interactions.
- Medium security, mobile RP: Basic mobile, Canvas, gyroscope, accelerometer, interactions.
- High security, web RP: Medium web, Mouse Dynamics, Keystroke dynamics, touchscreen dynamics, geolocation.
- High security, mobile RP: Medium mobile, touchscreen dynamics, environment, geolocation.

As mentioned before, these profiles are only a recommendation (i.e., not mandatory) and may be enhanced by using more or different attributes to meet specific RP requirements. If an RP chooses one of the profiles determined above and cannot have one or more of the required attributes, it is recommended to choose one of the immediate higher-level attributes. If this happens for the highest level, it is recommended to add at least two static attributes or one dynamic attribute to maintain an equivalent security level.

4.2. *Fingerprint generation*

The information gathered during this task has to be common to all end-users and as abundant as possible. The selected technology should be standard and multi-platform; the designed procedures have to be efficient (avoiding significant latency and resource consumption) and capable of obtaining accurate information. The gathering process needs to be connected to an RP storage system; it has to be quickly scalable (due to the volume of gathered data in average application domains) and, again, efficient. A storing procedure also needs to be defined, for example, to discard outdated, redundant or not useful information. Furthermore, entropy can be computed (Vastel et al., 2018b) to select representative data, excluding noisy data. Notice that this noisy information might be considered representative if the storing procedure does not discard any information over time. This entropy could also be used to determine a threshold useful for the next task in the proposed workflow.

It is essential to determine how the gathered information is going to be represented. For instance, Markov chains can be used if a specific user's behaviour is considered a state. In this case, a transition to an improbable state will be considered abnormal behaviour. Other techniques such as Symbolic Aggregate Approximation (SAX) can be used to represent temporal information in vectors of characters. Thus, each character represents a bunch of behavioural features over time. Then, comparing documents or strings techniques are used to determine the oddity of each sequence.

Our recommendation is to try first the simplest solutions, based on sequences of vectors, grouping the information into $n-grams$ in order to consider more information in a row. That is, to split each of the information vectors into sub-vectors of a given length. For example, given a vector containing 4 elements, they can be separated into subgroups of length 2 such that the first subgroup contains the first and second elements, the second subgroup contains the second and third elements and so on. Thus, each sub-vector contains information from the previous and the actual behavioural dynamics. This approach provides more flexibility to the RP when shaping the information to its specific needs than the mentioned before while consuming fewer resources.

4.3. *Modelling*

ML techniques focused on anomaly detection allow RPs to handle unlabelled data (unsupervised learning) and imbalanced data, the main challenges in this research. In the ML scope, these techniques are usually called outlier detection techniques.

These techniques are based on establishing a measure of distance or similarity (e.g. Euclidean distance and variations) between objects (Killourhy and Maxion, 2009). Notice that in our case, the objects are behavioural fingerprints. The defined distance is used to detect the most distant elements: if an object exceeds a fixed threshold, it is determined that it is an anomaly.

The ML algorithms that may be useful for an RP following the proposed workflow can be classified into three main groups (Chio and Freeman, 2018):

- Forecasting: They are based on classical supervised learning. In this group, the ML models are trained using previously gathered and labelled data. These algorithms try to perform predictions from the analysis of trends, seasons and cycles of time series. For example, ARIMA, Support Vector Regression, K Nearest Neighbour Regression, Multilayer Perceptron, or Bayesian Neural Network algorithms and variations fit in this scope.
- Unsupervised: These are focused on clustering the data into groups. The primary assumption is that normal behaviour will fit into the same clusters, while abnormal behaviours will be outside these clusters' bounds. Typically ML algorithms included in this scope are One-class Support Vector Machines (SVM), K-Means and Isolation forests.
- Density-based: This group is actually a sub-category of unsupervised learning, based on determining density regions. Expected points will probably fit a high-density region, while low-density regions will contain outliers (abnormal points). Some examples of algorithms in this group are Local Outlier Factor and Density-Based Spatial Clustering of Applications with Noise (DBSCAN).

A specific RP should analyse the gathered and stored data and the generated fingerprint: if these fingerprints are not labelled, unsupervised or density-based algorithms should be

selected; if labels (normal/abnormal) are available, forecasting algorithms could be suitable. Notice that labels might be unreliable, for example, if the information gathered for model training contains abnormal behaviours because there is an undetected security breach. This will cause that the models will consider abnormal behaviours as normal, making wrong predictions. Our recommendation is to use the gathered labels when possible and consider using unsupervised or density-based algorithms to verify and enhance the predictions. For instance, an interesting approach to improve the results combining different kinds of techniques is performing a first step of clustering (Shimshon et al., 2010). Categorising the analysed behaviour into specific groups enables applying different models, parameter values or thresholds in later steps.

All mentioned algorithms can be used in two different ways, offline and online. Offline models are often selected when using ML. They are considered to model the fingerprints if much information has been previously gathered and stored. An offline model is generated using historical data and is used over time to make predictions. This process can be repeated to re-train the model considering possible new behaviours in the fingerprints, obtaining a new offline model (Kang et al., 2007). This kind of model could be useful in use cases 1 and 3 (see Section 2.2).

It is worth mentioning the well-known problem of cold start. What happens till the model has enough information available to make predictions with reasonable accuracy? Note that when end-users start interacting with any RP, this RP does not have any previous behavioural information. The straightforward approach is to avoid this problem and start making predictions when it is considered that there is enough information to accomplish a good performance. Nevertheless, there are some approaches to address it differently. For example, using non-parametric models that do not need a training phase like Dynamic Time Warping (DTW) (Yan et al., 2018). In this case, two different models are used depending on the phase (i.e., cold start phase and stable phase). Another option is to group data from similar end-users to represent the new target user's behaviour for which the RP has not still information. Thus, the new end-user does not start from zero; instead, the RP considers other similar users' information (Cao and Lin, 2017). Note that a minimum quantity of behavioural dynamics of this new user is required to characterise him or her, but it is considerably less than the data required to use the proposed model.

On the other hand, online models are trained at the same time that they make their predictions. Therefore, these models can make predictions almost from the beginning, even for new end-users, improving their performance as the model receives more information over time.

The cold start problem may affect the three considered use cases. Our recommendation for an RP is to choose an online model for use case 2, whenever possible. This kind of model not only allows them to face the cold start problem, but they will also benefit from not having to define model re-training policies if the model's performance deteriorates at any moment. For use cases 1 and 3, the best option is to wait until the RP has enough behavioural information to start launching predictions. However, if it is necessary to face the cold start problem, the best option is to choose non-parametric models or models with short training phases to minimise as much as possible the time to start making accurate predictions.

4.4. *Evaluation*

When evaluating the modelling task results, it has to be considered that traditional evaluation metrics of ML models such as accuracy are not descriptive enough given the context of the proposed workflow. This is because an imbalanced problem needs to be addressed. For example, consider that the majority class contains the 99% of the data (normal behaviour). A toy ML model that always predicts a normal behaviour will obtain a 99% of accuracy; nevertheless, the abnormal behaviour will not be detected, making this model ineffective. The F1 score has demonstrated to be the right approach when addressing imbalanced problems (Lipton et al., 2014). This measure is the harmonic mean of the precision and the recall (also known as sensitivity). The precision represents the fraction of actual abnormal behaviour data among the predicted as abnormal by the model. The recall represents the fraction of the predicted as abnormal behaviour by the model data among the actual abnormal behaviour data. Thus, the F1 score combines both values into a unique metric to compare the performance of multiple models. Note that the F1 score assumes that the importance of the precision and the recall is the same (harmonic mean). This may not be suitable for all the issues addressed. For example, it can be assumed that in a critical infrastructure, the cost of making a mistake by letting an impostor through is not the same as the cost of asking a genuine user multiple times to authenticate in exchange for detecting more impostor users. In this example, the importance of recall is greater than the importance of precision (i.e., it is more important to ensure that no impostor authenticates than to make the system more usable but less secure).

Furthermore, specific metrics should be used to assess behavioural fingerprints, such as the False Acceptance Rate (FAR), the False Rejection Rate (FRR), and the Equal Error Rate (EER) (Eberz et al., 2017). These measures also take into account the fraction of the data of interest. The FAR is defined as the percentage of impostor users that the algorithm does not detect. The FRR is the percentage of legitimate users that the algorithm mismatch considering them impostors. Notice that different values of FAR and FRR can be obtained depending on the considered threshold. The EER is the optimal point in which the minimum FAR and FRR are obtained simultaneously for a specific threshold value. Thus, our recommendation is to use the EER when deciding the modelling task threshold and using the FAR and FRR to determine the obtained model's viability.

4.5. *Integration within identity management solutions*

The integration of a workflow as the one proposed in this work within identity federations implies different challenges.

The first, end-users might be reluctant to install/uninstall software if this will affect their privacy. The proposed workflow is a clear example in which users may sacrifice some privacy (depending on the selected fingerprint) to improve security. The proposed approach does not need to install any plugin or software, only the end-user consent for the specific RP

Table 2 – Summary of the proposed approach: UEBA workflow at RPs.

Task	Decision	Alternatives	Decision criteria
1. Fingerprint Selection	How to build unique session fingerprints: attributes or features	Static and dynamic attributes (Fig. 3)	Web or mobile RP, Target security level (basic, medium, high)
2. Fingerprint Generation	How to gather, store, pre-process and represent data: technologies, procedures and representation system	Web or mobile scripting technologies; SQL or No-SQL databases; Markov chains, SAX or n-grams	Efficiency, Accuracy, Resource consumption, Scalability
3. Modelling	How to detect anomalies in session fingerprints: techniques	Forecasting, unsupervised learning or density based algorithms; Offline or online models; Offline or online predictions	Selected fingerprint, Use case, Available resources at the RP
4. Evaluation	How to decide if more iterations of the workflow are required: metrics	Entropy of fingerprints, F1 score, FAR FRR and EER	Use case
5. Integration	How to use UEBA at the RP within the IAAA flow: modifications	Changes to specification (requests, tokens, flows); Changes to implementation (additional checks, data structure)	Use case, Allowable cost

and, perhaps, the deactivation of anti-fingerprinting countermeasures. It has to be highlighted that end-users may accept this scenario when working with an e-commerce site or with a banking app but not when working with a social network; the decision will be theirs in each case. It must be clear to the end-users that their fingerprint summarises features of their behaviour that enable the RP to detect anomalies, but they are required to have a keylogger or a similar spy software installed. It is also important to highlight that the only agent with access to these fingerprints will be the RP since they will be sent from the end-users devices with the corresponding encryption in the application layer (TLS).

On the other hand, acceptance among end-users will depend on the proposed solution efficiency and usability. The proposed approach needs to rely on a robust and unique fingerprint, not consuming user resources, avoiding unnecessary latency and false positives that would make users experience worse when accessing the RP.

IdPs will not have to install or deploy anything; all the integration burden will be on the RP side that it is the party interested in improving the levels of security offered to users. Therefore, they accept the resource consumption necessary to incorporate UEBA into their side of the IAAA flows.

The second challenge regards the proposed workflow acceptance among these RPs. This issue is two-fold. The acceptance within FIM ecosystems will depend on the modifications required to standard specifications or their implementation. The proposed approach tries to minimise these modifications, proposing them always as extensions that can be easily added to existing products and services by including (or becoming mandatory) specific parameters to requests, replies and tokens if possible.

5. Implementation and validation

A real application has been used to validate and evaluate the proposed approach, summarised in Table 2. Specifically, we are interested in implementing the whole workflow introduced in Fig. 2, exemplifying the specification of an RP-tailored fingerprint, assessing the behaviour of an anomaly detection technique and verifying the smooth integration of the proposed solution within OpenID Connect specification for the considered use cases.

5.1. *Use case description*

The RP is developed as a workplace chat app offering its users to authenticate using an external IdP to avoid creating and maintaining a local account.

The chat has been developed relying on a well-known open-source application called Letschat (2021). This chat is under the *MIT License*. The application runs on *Node.js* and *MongoDB*. It is worth mentioning that the application can be run in all the most popular browsers, making it suitable for use on both computers and smartphones. The RP has run during the experiments on an MSI laptop GF62 8RD-256XES equipped with an Intel Core i7 – 8750H (2.2 GHz, 9 MB). The RAM is 16 GB DDRIV (2666 MHz).

The IdP has been implemented with the open-source federation access management solution from *Forgerock* called OpenAM (2021). It has been deployed using a Docker image; therefore, only configuration issues have been addressed.

5.2. *Dataset and metrics*

A group of 11 different users (members of a research team with different age, gender and usage profiles) were using this application for one week to validate and evaluate the proposed approach as introduced in the following subsection. The generated data results in a complete UEBA dataset named “Keystroke and Mouse Dynamics for UEBA Dataset”. It is publicly available in the well-known Mendeley repository Keystroke and Mouse Dynamics for UEBA Dataset (2021).

The following metrics has been used to evaluate this first prototype:

- False Acceptance Rate (FAR): FP/(FP + TN)
- False Rejection Rate (FRR): FN/(FN + TP)

- Equal Error Rate (EER): Value of the intersection of FAR and FRR.

Where TP is the True Positives value (genuine users adequately authenticated by the system), TN are the True Negatives value (impostor users correctly kick out of the system), FP are the False Positives value (impostor users that are authenticated as genuine users by the system), and FN are the False Negatives value (genuine users that are incorrectly considered impostors and they are kicked out of the system). Notice that different FAR and FRR could be obtained for each ML model threshold (see Section 4.4). Thus, the global system could be more permissive or more restrictive, depending on the desired needs. The ideal system implies a trade-off for an optimal balance (a restrictive system will detect all impostors but will kick out most genuine users). The EER is defined as the optimum point that minimised both the FAR and the FRR simultaneously.

5.3. *First prototype*

5.3.1. *Fingerprint selection*

Following the guidelines of Section 4.1, we have a medium-security web RP. Nevertheless, the server from which the chat is offered is not very powerful, and potentially such an application could have many users. Therefore, the fingerprint should be kept as simple as possible to minimise resource consumption in the RP. However, it has to include the attributes that provide more information about the differences between users.

After a round of entropy tests, the user-agent and the screen resolution are selected among the static attributes, while keystroke and mouse dynamics are chosen among the dynamic. It has to be highlighted that, in a chat application, users will have to use both the keyboard and the mouse assiduously, so a large amount of data is expected to be collected.

5.3.2. *Fingerprint generation*

The technology selected at the RP to gather all the required data is *JavaScript*. Keyboard and mouse associated *JavaScript* events are used, particularly: keyup, keydown, mouseup, mousedown, mousemove, mouseup, mouseover, mouseenter, mouseout, click and dblclick.

The gathering component is based on the work (Leiva and Vivó, 2013). The component has been modified, improved and updated to meet all the requirements of the present work. It enables tracking all the proposed fingerprint attributes and storing them in a *MongoDB*. Events involving mouse dynamics have to be obtained by polling, gathered each 200 milliseconds due to the limited memory resources available and only if the position of the mouse changes.

Two central databases are created. The first one is named *EVTRACKINFO*. It contains information about the static attributes (e.g. user-agent and screen resolution). This database is linked to a second database: *EVTRACKTRACK*. This one stores the information about the behavioural dynamics (i.e. *JavaScript* events information).

A total of 347 and 142 691 records have been obtained for *EVTRACKINFO* and *EVTRACKTRACK*, respectively. Notice that, each record of *EVTRACKINFO* can be linked to multiple records of *EVTRACKTRACK*. The *EVTRACKTRACK* records are divided in 113 471 for keystroke dynamics and 29 220 for mouse dynamics. A mean of $28\,524 \pm 18\,541$ records has been obtained per user.

The typology of the data must be considered to generate a fingerprint for each independent user. Several transformations are applied to the data in order to extract significant features.

The static attributes do not need transformation; they are stored, straightly, as categorical variables. Dynamic attributes need some pre-processing instead.

Regarding the keystroke dynamics, a pressed key launches two *JavaScript* events: keyup and keydown. According to the previous works (Gamboa et al., 2007; Killourhy and Maxion, 2009) five main features can be extracted: the time between two keydown events (i.e. keydown-keydown), the releasing time (i.e. keydown-keyup), the time between a key is released, and the following key is pressed (i.e. keyup-keydown), the time between two keyup events (i.e. keyup-keyup), and finally, the specific character pressed.

In the case of mouse dynamics, six features have been extracted. The angle, distance and velocity features are obtained by grouping the mouse events in batches of two, generating one stroke (same as keyboard events). The elapsed time between events is also calculated. Finally, the type of events (i.e. mouseup, mousedown, etc.) associated with each of the two stroke elements is also collected.

With all this information, a specific fingerprint is generated for each user. This fingerprint contains three main components: the static features vector, the keystroke dynamics vector and the mouse dynamics vector.

5.3.3. *Modelling and evaluation*

The modelling and evaluating tasks are in charge of creating the ML models supporting UEBA and processing the new behavioural traces coming into the system. These tasks are considered together in this section because it would be back and forth between them multiple times until a robust model is reached (see Section 4.3). This makes the process more transparent and understandable because it is detailed precisely as it has been done. The software components implementing these tasks have been developed in *Python 3.7*. The *MongoDB* database, built during the previous task, feeds this component.

Experiments are focused on developing ML models for UEBA that enable a fair comparison between users' fingerprints. First, data from behaviours that can be considered normal is gathered. Then, the obtained dataset is split into train, test and validation sets. For each user, these sets contain 70%, 15%, and 15%, of the data, respectively. Notice that these sets correspond to actual user data.

The test and validation sets are then completed using other users' samples to simulate abnormal behaviour (impersonations, session hijacking). Thus, the final test and validation sets contain a similar number of genuine users and impostors. Summarising: the train set is used to model the data, the test set is used to verify the models' predictions and the validation set is used to determine if the models are robust and generalise properly.

Regarding the comparison of fingerprints, each component (static features, keystroke and mouse dynamics) is modelled independently. Thus, the diverse components can be activated or deactivated to complete a comparison of fingerprints.

A Naïve Bayes classifier is implemented to work with the static features. This classifier assumes that each variable is independent, and therefore, each variable determines a probability of belonging to a particular class (Ho and Kang, 2018). Due to the low number of participants in these first experiments (11), the obtained results show a perfect classification. The two features considered, user-agent and screen resolution, are enough to classify the participants. If the number of users increases, more static features should be considered to distinguish different users (following the recommendation given for medium security web RPs).

Regarding the keystroke dynamics, grouping data into *n – grams* has given good results in previous works (Zhao, 2006). Thus, concatenating 2, 3 and 6 vectors (character pressed features) is first considered. The Manhattan distance is used to model keystroke behavioural dynamics (Killourhy and Maxion, 2009).

First, the train set is used to get a mean vector of all the gathered information for each user. Next, the EER is used to select a threshold. In this case, a different mean vector and threshold are computed for each independent user. Thus, an independent model is obtained for each user.

The feature vectors are computed for each participant in the test set. Given the mean vector of a participant in the test set, normal behaviour is predicted when the Manhattan distance between that mean vector and the corresponding mean vector in the train set is lower than the corresponding threshold. Otherwise, abnormal behaviour is alerted.

Finally, with the same established threshold, the validation set is evaluated. The obtained results are discouraging, obtaining a mean EER of 0.511 ± 0.124. These adverse results are obtained because the state-of-the-art studies assumptions differ slightly from the ones of this research. In this work, a fair free behaviour model is required, meaning that users are free to interact with the application, while mentioned studies always gather the information in a restricted scenario accomplishing a set of very well defined tasks (for example, users transcript exact text multiple times).

Therefore, improvements are required to obtain optimal results. A more in-depth exploratory analysis of available data allows us to determine that vectors that start with the same character are similar. Thus, the keystroke vectors are clustered according to the first pressed character. Besides, the mean vector is not representative due to the dispersion of each user's data in this RP (a webchat).

Thus, the nearest neighbour approach using the Manhattan distance is proposed. Therefore, to predict a new participant as normal or abnormal, the nearest neighbour is computed. Then a threshold is fixed to minimise the EER for each user. Obtained results are shown in Table 3.

The obtained results show how this model represents better users' different behaviour, getting lower EER, FAR and FRR values.

Notice that these results compare each vector independently, but users interact with the RP generating a sequence of ordered vectors in real scenarios. Thus, following the proposed workflow, filling a temporary buffer of abnormality against time might be a useful approach.

Table 3 – Keystroke test and validation results for clustering by key technique using the Manhattan distance: FAR, FRR and EER with standard deviation.

	Test			Validation	
N-gram	FAR	FRR	EER	FAR	FRR
2	0.257	0.265	0.296 (0.099)	0.321	0.304
3	0.284	0.294	0.308 (0.099)	0.317	0.309
6	0.279	0.294	0.347 (0.141)	0.267	0.299

Table 4 – Keystroke test and validation results using the buffer and the Manhattan distance: FAR, FRR and EER with standard deviation.

	Test			Validation	
N-gram	FAR	FRR	EER	FAR	FRR
2	0.125	0.128	0.133 (0.107)	0.012	0.156
3	0.087	0.091	0.099 (0.126)	0.050	0.082
6	0.130	0.140	0.175 (0.156)	0.068	0.144

In this case, the train set is used to fill the buffer. For each vector, the distance to the threshold is computed. If the distance is greater than the threshold, then the value of the buffer increases. Otherwise, the value of the buffer decreases. The distance to the threshold is the value to be added or to be subtracted to the buffer. An exponential weighting is computed if a sequence of equal values (lower or higher) is observed. This makes the buffer fills or drowns faster. Then, a new threshold is determined for the buffer. This new threshold is established based on the minimum EER observed for the new values obtained applying the exponential weighting. Note that this threshold is established in the same way as the previous one.

The test set, which contains genuine and impostor data, is evaluated. In this particular case, all the impostor data information is concatenated at the end of the actual data. Results for these experiments are summarised in Table 4. This model outperforms the previous one. Besides, it is robust in the validation set. Notice that the results show that impostors can be detected quite accurately if the time in which a behavioural anomaly occurs is long enough.

However, the model is not suitable for our RP because, in a real scenario, abnormal behaviour can occur at any given time, not only at the end of a session, and the duration can be of a shorter time and can occur occasionally. Windowed vectors are defined to emulate a more realistic situation and solve these issues, grouping multiple vectors alongside. The windows size are fixed at 5, 10 and 20. For example, for the window size 10, a bunch of 10, normal vectors are concatenated to 10 impostor vectors. These resultant vectors are considered independent sessions. This process enables a fair comparison and allows us to evaluate the filling of the buffer. Obtained results are shown in Table 5 and Table 6. Notice that the results are more similar to those obtained in the previous case (ideal but not realistic scenario) when a bigger session size is consid-

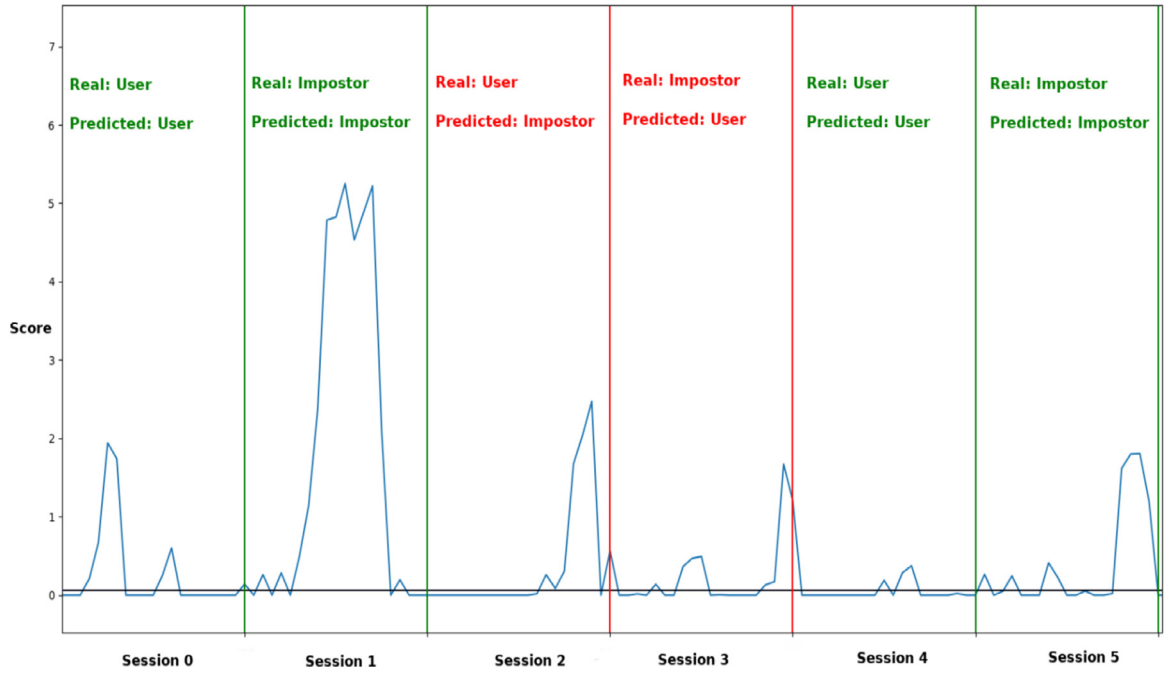

Fig. 4 – Excerpt of the buffer values for a specific user (keystroke dynamics).

Table 5 – Keystroke test results using the windowed buffer and the Manhattan distance.

Session size	N-gram	FAR	FRR	EER
5	2	0.221	0.213	0.238 (0.172)
5	3	0.254	0.242	0.205 (0.136)
5	6	0.235	0.197	0.247 (0.195)
10	2	0.169	0.162	0.279 (0.120)
10	3	0.185	0.194	0.188 (0.157)
10	6	0.221	0.228	0.206 (0.206)
20	2	0.141	0.162	0.128 (0.140)
20	3	0.172	0.140	0.145 (0.189)
20	6	0.153	0.172	0.174 (0.208)

Table 6 – Keystroke validation results using the windowed buffer and the Manhattan distance.

Session size	N-gram	FAR	FRR
5	2	0.199	0.220
5	3	0.213	0.213
5	6	0.262	0.211
10	2	0.170	0.216
10	3	0.170	0.170
10	6	0.155	0.196
20	2	0.126	0.188
20	3	0.161	0.110
20	6	0.155	0.180

Table 7 – Mouse dynamics test and validation results comparing each vector independently using One-class SVM: FAR, FRR and EER with standard deviation.

	Test			Validation	
N-gram	FAR	FRR	EER	FAR	FRR
1	0.447	0.448	0.455 (0.028)	0.440	0.441
2	0.442	0.443	0.427 (0.058)	0.444	0.447
3	0.440	0.440	0.432 (0.060)	0.442	0.445
6	0.447	0.419	0.407 (0.081)	0.476	0.468

ered. Fig. 4 shows the buffer values. The blue line represents the obtained score from the buffer at a given time. The black horizontal line is the determined threshold. Green lines represent that a correct prediction is accomplished (interchangeably for normal or abnormal behaviour). Red lines represent a wrong prediction.

The obtained results show that the best performance is obtained using $2-gram$ and $3-gram$. As expected, when the session is larger, better results can be obtained because more data are analysed in an independent window.

Finally, the same assumptions and processes that have succeeded when analysing keystroke dynamics can be applied to analyse mouse dynamics. First, the Manhattan distance is assessed. In this case, the selection of $n-grams$ is 1, 2, 3 and 6. An independent comparison between vectors is performed, obtaining discouraging results (EER of 0.496 ± 0.160) for the best parameters ($6-gram$). Furthermore, outliers detection techniques are evaluated. Specifically, One-class SVM is tested in the same scenario. In this case, obtained results are slightly better than those obtained with the Manhattan distance, but they do not reach the keyboard dynamics levels (see Table 7). Thus, the Manhattan distance is discarded.

Following the guidelines for keystroke dynamics, a buffer of abnormality is also considered. During the first group of experiments, the impostor data is concatenated at the end of the

Table 8 – Mouse dynamics test and validation results using the buffer and the One-class SVM: FAR, FRR and EER with standard deviation.

	Test			Validation	
N-gram	FAR	FRR	EER	FAR	FRR
1	0.072	0.108	0.151 (0.083)	0.115	0.093
2	0.205	0.214	0.249 (0.162)	0.168	0.088
3	0.399	0.399	0.409 (0.081)	0.306	0.173
6	0.415	0.447	0.399 (0.088)	0.460	0.479

Table 9 – Mouse dynamics test results using the windowed buffer and One-class SVM.

Session size	N-gram	FAR	FRR	EER
5	1	0.352	0.357	0.373 (0.071)
5	2	0.376	0.388	0.337 (0.131)
5	3	0.376	0.360	0.381 (0.076)
5	6	0.366	0.386	0.296 (0.175)
10	1	0.237	0.225	0.242 (0.131)
10	2	0.292	0.306	0.279 (0.120)
10	3	0.329	0.325	0.327 (0.119)
10	6	0.285	0.381	0.278 (0.181)
20	1	0.151	0.151	0.143 (0.122)
20	2	0.232	0.232	0.220 (0.135)
20	3	0.274	0.246	0.240 (0.205)
20	6	0.258	0.366	0.248 (0.188)

Table 10 – Mouse dynamics validation results using the windowed buffer and One-class SVM.

Session size	N-gram	FAR	FRR
5	1	0.353	0.355
5	2	0.381	0.410
5	3	0.430	0.426
5	6	0.425	0.451
10	1	0.307	0.329
10	2	0.341	0.372
10	3	0.370	0.380
10	6	0.386	0.462
20	1	0.256	0.277
20	2	0.295	0.356
20	3	0.355	0.385
20	6	0.363	0.368

actual data. Obtained results are summarised and verified in Table 8.

To conclude, the addition of information about the session (using the windowed buffer) is considered too. Results are shown in Tables 9 and 10. The buffer is shown in Fig. 5. For the presented case, the threshold for the buffer is very restrictive. Notice that the system predicts an impostor for session 2 despite the low buffer values. This issue makes the system more secure (it detects impostors more often), but it also kicks out legitimate users more often.

Summarising, the best overall results for this specific RP (a webchat) are obtained when modelling keystroke dynamics. In particular, the best results for a real scenario are obtained using the Manhattan distance and computing the nearest neighbours with $3-gram$ and sessions of 20 vectors. On the other hand, for mouse dynamics, the best results are obtained using a One-class SVM with $1-gram$ and sessions of 20 vectors.

5.4. *Integration within identity management solutions*

Fig. 6 shows the OpenID Connect flow used by our application. As it can be observed, it involves a request from the RP to the IdP (step 1), the authentication of the end-user at the IdP (steps 2 to 3) or at least, her consent, and the redirection of an authorisation code to the RP (steps 4 and 5). The RP uses this code to retrieve the ID token and Access token (steps 6 and 7).

This integration of the proposed approach in this workflow depends on the use case in question. If we consider the ones mentioned in Section 2.2:

- Use case 1: This use case requires only a slight modification to the current FIM specifications. It involves making an optional parameter of the Authentication Request, *acr_values*, mandatory. The RP uses it to specify the LoA it requires from the end-user authentication process at the IdP. The higher the risk associated with access (because it has been detected that it is not normal), the higher the LoA required. Similarly, the *acr* parameter of the ID token becomes mandatory. The IdP uses it to inform the RP of the method used to authenticate the end-user with the requested LoA. Specific implementations will handle differently the methods available at the IdP and how the RP reacts if, for example, the IdP cannot reach the required LoA. This no longer affects the specification; it can be solved at the code level.
- Use case 2: RPs are allowed by the current specifications to make a Token Request at the IdP by presenting its authorisation code to the Token Endpoint using the value of the *authorisation_code* obtained in step 5 of Fig. 6. Therefore, this use case does not require a modification of the current specifications. Again, it requires an improvement in the RP implementation: a Token Request will be forced every time the continuous authentication solution concludes that the legitimate user did not perform the login or that the current session has been hijacked at some point.
- Use case 3: This use case does not affect the traditional federated IAAA flows; it also implies new management and communication procedures at the RP to be implemented if impersonation is suspected.

The only change to the current OpenID Connect specification required to integrate it with the proposed approach would be turning two parameters (one in the Authentication Request and one in the ID token) into mandatory. The minimisation of mandatory changes to current specifications dramatically facilitates their adoption by different providers and RPs.

5.5. *Performance and security analysis*

In previous sections, it has been mentioned that UEBA techniques can support different use cases at RPs belonging to different application domains running on very heterogeneous platforms. This is the reason why resource consumption

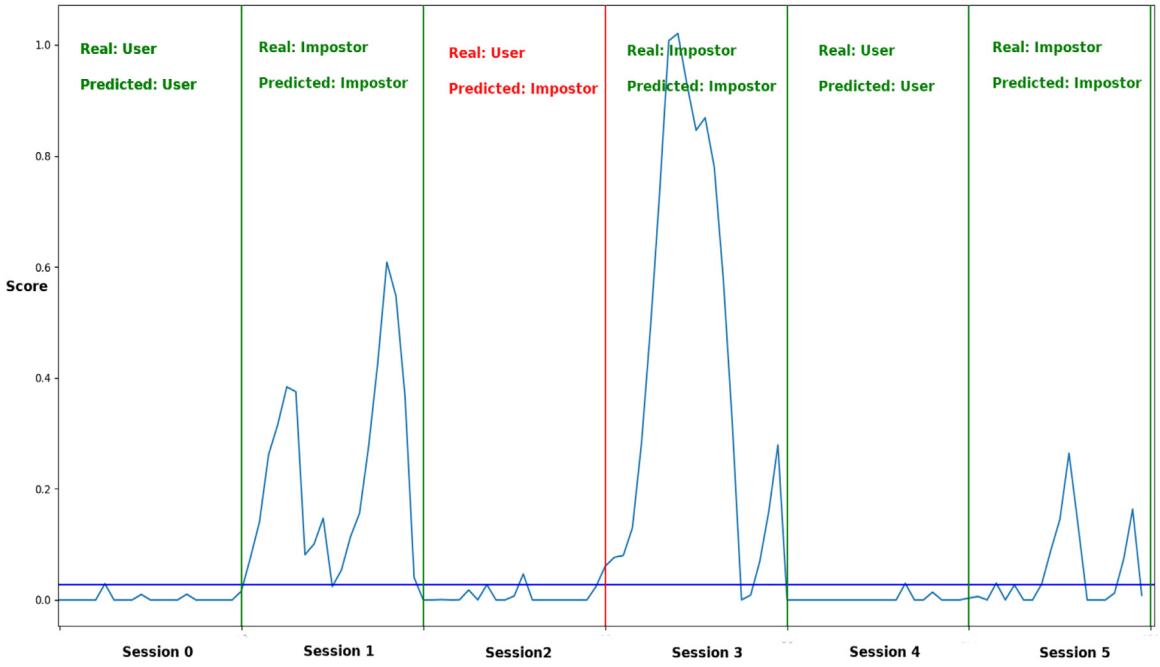

Fig. 5 – Excerpt of the buffer values for a specific user (mouse dynamics).

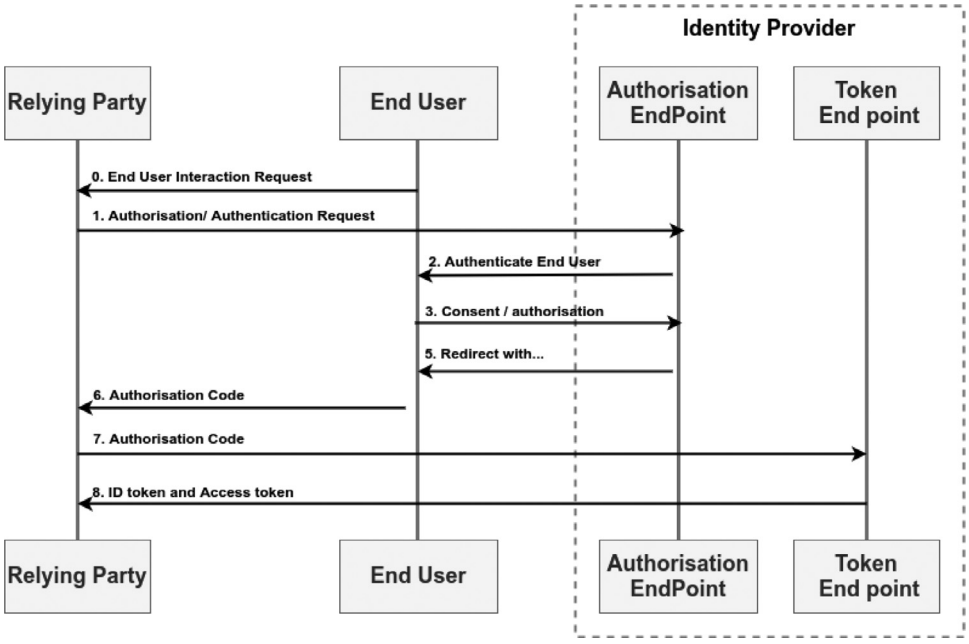

Fig. 6 – Authorisation code flow with OpenID Connect.

should be minimised (it has been mentioned during the Fingerprint Selection task), and latency should be kept at values that do not affect the regular operation of the RP (e.g. in case of use 2) or the end-user quality of experience (e.g. in case of use 1).

The average execution time to compare two vectors using nearest neighbours and Manhattan distances has been 0.000416 ± 0.000883 nanoseconds in our experiments with the RP resources. In the case of the One-class SVM, the average execution time has been 0.000040 ± 0.0000923 nanoseconds. These values allow us to use UEBA techniques in all the mentioned use cases without affecting user experience, even in pre-authentication or online scenarios.

The last group of performed experiments is devoted to analysing the proposed approach's security improvement in the most demanding use case, the number 2 (continuous authentication). Two different kinds of attacks have been performed. First, credential theft and spoofing. Participants publish their credentials, and they have to impersonate others using their own devices. Second, session hijacking. In this case,

Table 11 – Results for the credential theft/spoofing attack in the use case 2.

Type	FAR	FRR
Static	0	0
Keystroke	0.104	0.082
Mouse	0.146	0.127

Table 12 – Results for the session hijacking attack in the use case 2.

Type	FAR	FRR
Static	1	1
Keystroke	0.149	0.104
Mouse	0.175	0.168

participants have to impersonate others using the device of the victim. The models integrated with the web chat application are those with the best performance in the previous subsection.

Results obtained with the first attack are shown in Table 11. An average of 12.2 ± 4.176 interactions are needed to detect an impostor from the keystroke dynamics. On the other hand, an average of 18.8 ± 2.081 interactions is required to detect an impostor from mouse dynamics. As it was mentioned in the previous subsection, the results obtained from the static features imply a perfect classification. Nevertheless, worse results are expected in a real scenario with more participants.

Regarding the second attack, session hijacking using the victim's device, results are summarised in Table 12. For the keystroke dynamics, an average of 13.1 ± 4.223 interactions are needed to detect an impostor, while an average of 18.1 ± 2.714 interactions are needed regarding the mouse dynamics. Notice that, in this case, the static features are not useful at all (because the impostor is using the victim's device).

It can be observed that the developed models have better performance detecting the first attack than the second. Notice that it is more difficult to detect an impersonation when it is performed from the own victim's device. Static attributes are not helpful in this case. Furthermore, dynamic attributes, consistently abnormal for a short period, in the beginning, tend to normalise after this period because using the same keyboard and mouse generates similar behaviour from the impostor that from the victim.

It has to be pointed out that this makes it more difficult to detect abnormal behaviour, even though the results are entirely consistent and valuable for the considered use cases.

6. Conclusions

This paper has proposed a workflow allowing Relying Parties within identity federations to perform User and Entity Behaviour Analytics to raise their end-users security levels. The proposed approach is based on building session fingerprints from users or entities' behaviour data (static and dynamic attributes), applying anomaly detection techniques to detect impersonation due to credential theft or session hijacking. The alerts generated from these techniques can be integrated into different ways with the flows that FIM schemes specify as standard: pre or post-authentication, online or offline.

Analysis performed within a real use case using a workspace chat application and OpenID Connect shows how the proposed approach allows RPs to count with accurate (obtaining suitable FAR and FRR values given the potential use cases), easy to adopt (implying very slight modifications to current FIM specifications) and flexible (allowing performing UEBA at different application domain RPs, in different use cases) solutions.

Performed experiments in the considered use case have shown that no one solution fits all RPs. Each one will have to decide which fingerprint best summarises its users' behaviour (entities), then which models allow anomaly detection based on these fingerprints given the available storage, computing resources, and the required latency due to the specific application domain.

Alternatives to building privacy-preserving fingerprints capable of capturing the user's behaviour to detect anomalies without revealing any sensitive information are being explored.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

CRediT authorship contribution statement

Alejandro G. Martín: Software, Validation, Formal analysis, Investigation, Data curation, Writing - original draft. **Marta Beltrán:** Conceptualization, Validation, Investigation, Writing - review & editing, Supervision, Funding acquisition. **Alberto Fernández-Isabel:** Methodology, Validation, Formal analysis, Visualization. **Isaac Martín de Diego:** Methodology, Validation, Formal analysis, Visualization.

Acknowledgements

Research supported by grants from the Education, Youth and Sports Council of the Comunidad de Madrid and the European Social Fund of the European Union (ref. PEJ-2017-AI/TIC-6403).

REFERENCES

Abuhamad M, Abusnaina A, Nyang D, Mohaisen D. Sensor-based continuous authentication of smartphones users using behavioral biometrics: a contemporary survey. IEEE Internet Things J. 2020;8(1):65–84.

Ahmed AAE, Traore I, Ahmed A. Digital fingerprinting based on keystroke dynamics.. In: HAISA; 2008. p. 94–104.

Bakar KAA, Haron GR. Adaptive authentication based on analysis of user behavior. In: 2014 Science and Information Conference. IEEE; 2014. p. 601–6.

Beltrán M. Identifying, authenticating and authorizing smart objects and end users to cloud services in internet of things. Comput. Secur. 2018;77:595–611.
Bezawada, B., Bachani, M., Peterson, J., Shirazi, H., Ray, I., Ray, I., 2018. Iotsense: behavioral fingerprinting of IoT devices. arXiv preprint arXiv:1804.03852
Bhana B, Flowerday S. Passphrase and keystroke dynamics authentication: usable security. Comput. Secur. 2020;96.
Bhatnagar M, Jain RK, Khairnar NS. A survey on behavioral biometric techniques: mouse vs keyboard dynamics. Int. J. Comput. Appl 2013;975:8887.
Cao H, Lin M. Mining smartphone data for app usage prediction and recommendations: a survey. Pervasive Mob. Comput. 2017;37:1–22.
Chadwick DW. Federated identity management. In: Foundations of Security Analysis and Design V. Springer; 2009. p. 96–120.
Chandola V, Banerjee A, Kumar V. Anomaly detection: a survey. ACM Comput. Surv. (CSUR) 2009;41(3):15.
Chio C, Freeman D. Machine Learning and Security: Protecting Systems with Data and Algorithms. "O'Reilly Media, Inc."; 2018.
Chow R, Jakobsson M, Masuoka R, Molina J, Niu Y, Shi E, Song Z. Authentication in the clouds: a framework and its application to mobile users. In: Proceedings of the 2010 ACM Workshop on Cloud Computing Security Workshop; 2010. p. 1–6.
Eberz S, Rasmussen KB, Lenders V, Martinovic I. Evaluating behavioral biometrics for continuous authentication: challenges and metrics. In: Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security; 2017. p. 386–99.
Eckersley P. How unique is your web browser?. In: International Symposium on Privacy Enhancing Technologies Symposium. Springer; 2010. p. 1–18.
Formby D, Srinivasan P, Leonard A, Rogers J, Beyah RA. Who's in control of your control system? Device fingerprinting for cyber-physical systems.. In: Proceedings of the 23rd Annual Network and Distributed System Security Symposium, NDSS. The Internet Society; 2016. p. 1–15.
de Fuentes JM, Gonzalez-Manzano L, Ribagorda A. Secure and usable user-in-a-context continuous authentication in smartphones leveraging non-assisted sensors. Sensors 2018;18(4):1219.
Gamboa H, Fred A, Jain A. Webbiometrics: user verification via web interaction. In: 2007 Biometrics Symposium. IEEE; 2007. p. 1–6.
Gascon H, Uellenbeck S, Wolf C, Rieck K. In: Sicherheit 2014–Sicherheit, Schutz und Zuverlässigkeit. Continuous authentication on mobile devices by analysis of typing motion behavior; 2014.
Gómez-Boix A, Laperdrix P, Baudry B. Hiding in the crowd: an analysis of the effectiveness of browser fingerprinting at large scale. In: Proceedings of the 2018 World Wide Web Conference; 2018. p. 309–18.
Gu X, Yang M, Zhang Y, Pan P, Ling Z. Fingerprinting network entities based on traffic analysis in high-speed network environment. Secur. Commun. Netw. 2018;2018. doi:10.1155/2018/6124160.
Herrmann D, Fuchs K-P, Federrath H. In: Sicherheit 2014–Sicherheit, Schutz und Zuverlässigkeit. Fingerprinting techniques for target-oriented investigations in network forensics; 2014.
Ho J, Kang D-K. One-class naïve Bayes with duration feature ranking for accurate user authentication using keystroke dynamics. Appl. Intell. 2018;48(6):1547–64.
Hodge V, Austin J. A survey of outlier detection methodologies. Artif. Intell. Rev. 2004;22(2):85–126.
Huda S, Abawajy J, Al-Rubaie B, Pan L, Hassan MM. Automatic extraction and integration of behavioural indicators of malware for protection of cyber–physical networks. Future Gener. Comput. Syst. 2019;101:1247–58.
IETF,. The oauth 2.0 authorization framework. https://tools.ietf.org/html/rfc6749. Accessed: 2021-01-13.
Ikuesan AR, Venter HS. Digital behavioral-fingerprint for user attribution in digital forensics: are we there yet? Digit. Investig. 2019;30:73–89.
Kang P, Hwang S-s, Cho S. Continual retraining of keystroke dynamics based authenticator. In: International Conference on Biometrics. Springer; 2007. p. 1203–11.
Keystroke and Mouse Dynamics for UEBA Dataset, Mendeley Data, v2. https://doi.org/10.17632/f78jsh6zp9.2. Accessed: 2021-01-13.
Killourhy KS, Maxion RA. Comparing anomaly-detection algorithms for keystroke dynamics. In: 2009 IEEE/IFIP International Conference on Dependable Systems & Networks. IEEE; 2009. p. 125–34.
Lackner G, Teufl P, Weinberger R. User tracking based on behavioral fingerprints. In: International Conference on Cryptology and Network Security. Springer; 2010. p. 76–95.
Laperdrix P, Bielova N, Baudry B, Avoine G. Browser fingerprinting: a survey. ACM Trans. Web (TWEB) 2020;14(2):1–33.
Laperdrix P, Rudametkin W, Baudry B. Beauty and the beast: diverting modern web browsers to build unique browser fingerprints. In: 2016 IEEE Symposium on Security and Privacy (SP). IEEE; 2016. p. 878–94.
Leiva LA, Vivó R. Web browsing behavior analysis and interactive hypervideo. ACM Trans. Web (TWEB) 2013;7(4):1–28.
let's chat, https://sdelements.github.io/lets-chat. Accessed: 2021-01-13.
Li D, Deng L, Liu W, Su Q. Improving communication precision of IoT through behavior-based learning in smart city environment. Future Gener. Comput. Syst. 2020;108:512–20.
Lipton ZC, Elkan C, Narayanaswamy B. Thresholding classifiers to maximize F1 score. Mach. Learn. Knowl. Discov. Databases 2014;8725:225–39.
Meng W, Li W, Wang Y, Au MH. Detecting insider attacks in medical cyber–physical networks based on behavioral profiling. Future Gener. Comput. Syst. 2020;108:1258–66.
Meng W, Wang Y, Wong DS, Wen S, Xiang Y. Touchwb: touch behavioral user authentication based on web browsing on smartphones. J. Netw. Comput. Appl. 2018;117:1–9.
Mondal S, Bours P. Combining keystroke and mouse dynamics for continuous user authentication and identification. In: 2016 IEEE International Conference on Identity, Security and Behavior Analysis (ISBA). IEEE; 2016. p. 1–8.
Navas J, Beltrán M. Understanding and mitigating openid connect threats. Comput. Secur. 2019;84:1–16.
Oasis. Security assertion markup language 2.0. http://saml.xml.org/saml-specifications. Accessed: 2021-01-13.
OIDF. OpenID Connect 1.0. http://openid.net/connect/. Accessed: 2021-01-13.
OpenAM, https://backstage.forgerock.com/docs/openam/13.5/getting-started/. Accessed: 2021-01-13..
Sato H, Kanai A, Tanimoto S, Kobayashi T. Establishing trust in the emerging era of IoT. In: 2016 IEEE Symposium on Service-Oriented System Engineering (SOSE). IEEE; 2016. p. 398–406.
Sciancalepore S, Piro G, Caldarola D, Boggia G, Bianchi G. Oauth-IoT: an access control framework for the internet of things based on open standards. In: 2017 IEEE Symposium on Computers and Communications (ISCC). IEEE; 2017. p. 676–81.
Shahid MR, Blanc G, Zhang Z, Debar H. IoT devices recognition through network traffic analysis. In: 2018 IEEE International Conference on Big Data (Big Data). IEEE; 2018. p. 5187–92.
Shen C, Li Y, Chen Y, Guan X, Maxion RA. Performance analysis of multi-motion sensor behavior for active smartphone

authentication. IEEE Trans. Inf. Forensics Secur. 2017;13(1):48–62.
Shimshon T, Moskovitch R, Rokach L, Elovici Y. Clustering di-graphs for continuously verifying users according to their typing patterns. In: 2010 IEEE 26-th Convention of Electrical and Electronics Engineers in Israel. IEEE; 2010. p. 000445–9.
Smith-Creasey M, Rajarajan M. A novel word-independent gesture-typing continuous authentication scheme for mobile devices. Comput. Secur. 2019;83:140–50.
Taneja M. An analytics framework to detect compromised IoTdevices using mobility behavior. In: 2013 International Conference on ICT Convergence (ICTC). IEEE; 2013. p. 38–43.
Thangavelu V, Divakaran DM, Sairam R, Bhunia SS, Gurusamy M. Deft: a distributed IoT fingerprinting technique. IEEE Internet Things J. 2018;6(1):940–52.
Vastel A, Laperdrix P, Rudametkin W, Rouvoy R. FP-scanner: the privacy implications of browser fingerprint inconsistencies. In: 27th {USENIX} Security Symposium ({USENIX} Security 18); 2018. p. 135–50.
Vastel A, Rudametkin W, Rouvoy R. FP-tester: automated testing of browser fingerprint resilience. In: 2018 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW). IEEE; 2018. p. 103–7.
Voris J, Song Y, Salem MB, Hershkop S, Stolfo S. Active authentication using file system decoys and user behavior modeling: results of a large scale study. Comput. Secur. 2019;87:101412.
Xiaofeng L, Shengfei Z, Shengwei Y. Continuous authentication by free-text keystroke based on CNN plus RNN. Procedia Comput. Sci. 2019;147:314–18.
Yan J, Qi Y, Rao Q, Qi S. Towards a user-friendly and secure hand shaking authentication for smartphones. In: 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE). IEEE; 2018. p. 1170–9.
Yang L, Zhi Y, Wei T, Yu S, Ma J. Inference attack in android activity based on program fingerprint. J. Netw. Comput. Appl. 2019;127:92–106.
Zhao Y. Learning user keystroke patterns for authentication. Proc. World Acad. Sci. Eng. Technol. 2006;14:65–70.

Alejandro G. Martín received the B.E in Software Engineering from Rey Juan Carlos University (URJC) in 2016, the M.S in Data Science from URJC in 2017 and the M.S degree in Cyber Security from Oberta Catalunya University (UOC) in 2019. He is currently a Ph.D. student of the Higher Technical School of Computer Engineering (ETSII) at URJC. His research interests include cyber security, identity management, user and entity behavior analytics, data science, machine learning and text mining.

Marta Beltrán received the master's degree in Electrical Engineering from Universidad Complutense of Madrid (Spain) in 2001, the master's degree in Industrial Physics from UNED (Spain) in 2003 and the Ph.D. degree from the Department of Computing, Universidad Rey Juan Carlos, Madrid (Spain) in 2005. She is currently working with this department as an Associate Professor. She is the leader of the GAAP research group, co-founder of the Cybersecurity Cluster and she has published extensively in high-quality national and international journals and conference proceedings in the areas of computer security and privacy, and parallel and distributed systems. Her current research interests are Cloud computing, Edge/Fog Computing and Internet of Things, specifically, Identity and Access Management and privacy-preserving mechanisms for these paradigms.

Dr. Alberto Fernández Isabel Professor and postdoctoral researcher of the Higher Technical School of Computer Engineering (ETSII) at Rey Juan Carlos University (URJC). Master in Data Science Professor. Ph.D. in Computer Science and Master's Degree in Research in Artificial Intelligence at Complutense University of Madrid (UCM). Research interests: intelligent agents, data visualization and natural language processing. Application domains: distributed programming, sentiment analysis, agent-based collaboration and negotiation, smart cities and simulations.

Isaac Martín de Diego Ph.D. in Mathematical Engineering at University Carlos III of Madrid (UC3M). Extraordinary Doctorate Award. Associated Professor of the Higher Technical School of Computer Engineering (ETSII) at Rey Juan Carlos University (URJC). Co-founder of the URJC Data Science Laboratory. Head of the URJC Master in Data Science. Secretary of the Academic Council of the ERICSSON Chair on Data Science applied to 5G. His research interests include methods, processes and tools for Data Science in various application domains: artificial vision, opinion mining, security and biostatistics, with special interest on Machine Learning algorithms and combination of information methods.