The Measurable Environment as Nonintrusive Authentication Factor on the Example of WiFi Beacon Frames

Philipp Jakubeit[1(✉)], Andreas Peter[1,2], and Maarten van Steen[1]

[1] University of Twente, Drienerlolaan 5, 7500 AE Enschede, The Netherlands
{p.jakubeit,a.peter,m.r.vansteen}@utwente.nl
[2] University of Oldenburg, Ammerländer Heerstraße 114-118,
26129 Oldenburg, Germany
andreas.peter@uol.de

Abstract. We explore a method to fingerprint a location in terms of its measurable environment to create an authentication factor that is nonintrusive in the sense that a user is not required to engage in the authentication process actively. Exemplary, we describe the measurable environment by beacon frames from the WiFi access points in the user's proximity. To use the measurable environment for authentication, measurements must be sufficiently discriminating between locations and similar at the same location. An authentication factor built from the measurable environment allows us to describe a user's location in terms of measurable signals. Describing a location in terms of its measurable signals implies that we do not require an actual geographical mapping of the user's location; comparing the measured signals is sufficient to create a location-based authentication factor. Only recognizing an earlier observed environment distinguishes our approach from other location-based authentication factors. We elaborate on using signals in the user's environment in the background without user involvement to create a privacy-preserving but nonintrusive authentication factor suitable for integration into existing multi-factor authentication schemes.

1 Introduction

Multi-factor authentication schemes are the de facto standard when it comes to user authentication. *Authentication* is the 'provision of assurance that a claimed characteristic of an entity is correct' [11]. Authentication factors are conceptually grouped according to the characteristic claimed: knowledge, possession, biometry, and location [2]. The standard in multi-factor authentication schemes in 2022 is two-factor authentication. When authenticating to a particular service, the first factor is typically based on a username-password combination. Only the legitimate user is assumed to know this combination (knowledge-based authentication claim). The second authentication factor is usually a token sent to the user's registered device (SMS, Authenticator app) to strengthen the assurance

A. Saracino and P. Mori (Eds.): ETAA 2022, LNCS 13782, pp. 48–69, 2023.
https://doi.org/10.1007/978-3-031-25467-3_4

that the user is who he claims to be. The user must retype or confirm this token to the service to get authenticated (possession-based authentication claim). We aim to extend this scheme by taking the user's location into account. Extending implies that the knowledge-based and possession-based authentication factors remain parts of the scheme. We add location as a third authentication factor to the two-factor scheme. Location as a claim characteristic might even replace the second factor in certain situations.

The advantage of a location-based authentication claim is that the characteristic claimed does not require the user's active involvement. The location is about the whereabouts of a user, not the user himself. To not require a user to be actively involved in the authentication process is called *nonintrusive authentication* [16]. Nonintrusiveness allows the user to be undisturbed and the service to adjust the remote trust [22] that the user is whom he claims to be. Nonintrusiveness also solves the problem of infrequent authentication [22], the shortcoming that authentication only occurs at the beginning of a session. In other words, using a user's location as an authentication factor allows the service to probe the user for his authentication claim at any time during the session.

The disadvantage of location as an authentication factor is twofold. On the one hand, the claimed characteristic is not of the specific user but the user's environment. On the other hand, a user's location is privacy-sensitive information as it maps the whereabouts of the user when using traditional means to describe the user's location (e.g., GPS or IP subnet ranges).

We asked ourselves how far these disadvantages are necessary for location-based authentication and how we can avoid them. These disadvantages provide us also with an insight into privacy sensitivity. Intuitively, a service only needs to validate a claimed characteristic of the location. Therefore, we do not need to consider where a location is in terms of geographical mapping (like with GPS or IP addresses). We only need to validate the claimed characteristic of the location. To explore this further, we choose the *measurable environment* (ME) as the claimed characteristic of a location.

The ME consists of electromagnetic signals in our surroundings. Appropriate sensors can measure these signals. In the following, we investigate whether it is possible to fingerprint a user's measured environment. A successful fingerprinting scheme of the ME allows us to compare MEs, distinguish different environments and recognize similar environments. For the purpose of authentication, we compare a measured environment of a location to an earlier observed measurement of the location and compute the similarity of measurements.

We envision the ME as an authentication factor to extend the described standard two-factor authentication scheme to a three-factor authentication scheme which assures knowledge, possession, and consistency of the ME. After the username and password prompt, the service can decide to use the second factor and the ME or even replace the second factor with the ME during the login phase. The service checks whether it recognizes the newly observed ME of the user and uses this claim to assure the user's authenticity. Such an authentication factor allows the service to assure consistency of the user's ME, not just during the

login time but continuously. The ME as an authentication factor is only limited by the ME's availability, the user's bandwidth, and the time it takes to fingerprint a location in terms of its ME.

We choose WiFi to fingerprint the ME as it is a ubiquitous signal, and the vast majority of devices have sensors to receive WiFi signals. We construct an ambient WiFi fingerprint from beacon frames emitted by WiFi access points (APs) to indicate their presence. We measure these emitted beacon frames for a particular duration at a specific location to construct a fingerprint.

In the remainder of this paper, we discuss how an ME can be fingerprinted and classified before presenting our WiFi instantiation. We continue evaluating the performance of our classification of an ME in terms of WiFi beacon frames. Having a working fingerprinting mechanism, we focus on integrating the WiFi fingerprints into an authentication scheme and discuss the security of authentication based on the ME. We compare our results with results from the literature and close with a discussion, including paths for future works and our conclusions.

2 The Measurable Environment (ME)

We assume the ME as a noisy source, conceptually similar to biometric features of an individual [15] or unique hardware features [7] of devices. We further assume that some ME will be present at most locations. Using a specific sensor, we can pick up a measurement of the ME at a specific location and time. We will position a sensor at certain locations to conduct measurements during specific times. We use these fixed measurements to conduct and evaluate our classification.

2.1 Illustration of the ME

The signals and the sensor must be available for a measured environment to be suitable for fingerprinting. If signals and sensors are available, the ME must have two additional properties to be viable for fingerprinting: measurements of different MEs must be different, and measurements of the same ME must be similar over time. Both need to be evaluated on a per sensor and per signal basis. In Fig. 1, we show three measured environments which we mapped to the two-dimensional space for illustrative purposes only. Figure 1 shows by its visible clusters that measurements of the same ME taken at different times are similar. We also observe that the different locations have no overlap in their measured MEs.

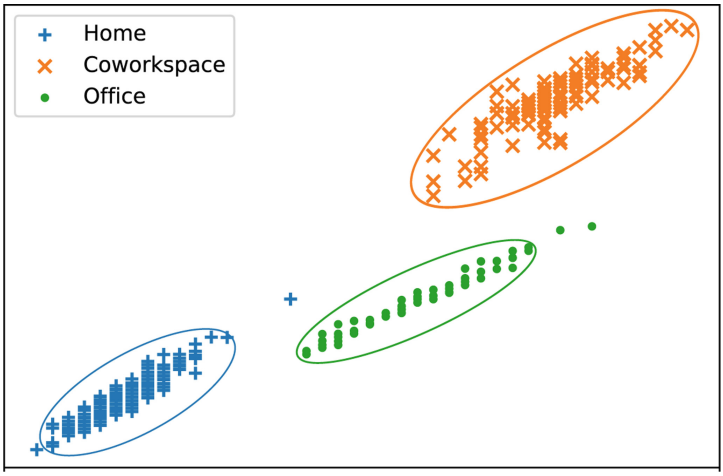


Fig. 1. Illustration of measurements taken at three different MEs: at home, at work, and in a coworking space. Each point represents a fingerprint of a specific ME of one second. The distance between the points describes their similarity. The clusters of points represent a template of an ME, and the boundaries surrounding the clusters represent the threshold of this specific template. If a newly obtained fingerprint falls within the boundaries, we classify the fingerprint as taken at the same ME associated with the template. The figure further shows that the distance between some fingerprints of the ME in the office and the home case fall outside their boundary. These outliers represent false negatives in a classification.

2.2 Fingerprinting the ME

We assume a *sensor* s, which is capable of measuring the environment, taking a *measurement* m. For this work, s is fixed at a specific location. Two measurements m and m' of the same ME are unlikely to be identical because physical-environment data is measured, which is susceptible to interference. To get a more detailed description of the ME, we take multiple measurements of the same ME into account. We build a *fingerprint* $\mathbf{F}$ from a set of measurements by a single sensor s during a time window of t seconds. The associated sensor is denoted as $s(\mathbf{F})$, and the duration of the set of measurements as $t(\mathbf{F})$. To be capable of recognizing a fingerprint, we construct a *template* $\mathbb{F}$ from a set of n fingerprints measured by a single sensor s at the same ME. The associated sensor is denoted as $s(\mathbb{F})$. In Fig. 1 this correlates to taking a cluster of n points as the template.

2.3 Classifying Fingerprints of MEs

When measuring an environment, we want to check if we are dealing with an ME we observed earlier. That means we require a similarity metric and a decision rule to derive whether two fingerprints are sufficiently similar.

In Fig. 1, the distance of one point toward a cluster of points describes their similarity. The elliptic boundary around a cluster of points forms the threshold

for a decision rule. A new observation that falls within the boundary would classify as being at the same ME. In contrast, an observation outside this boundary would classify as remote to that ME. We want to determine these boundaries around a cluster to create a meaningful decision rule. To prevent overfitting, we want to do this in such a way that the boundaries are neither too small (resulting in too many false negatives) nor do we want too large boundaries (resulting in too many false positives). Therefore, we base the threshold on data from within the boundary and data from outside. Suppose the data from within is all we have. As the outliers in Fig. 1 indicate, the boundary would easily be chosen as too large. The boundary will be much tighter if we use data from outside, i.e., from other remote MEs. A tight boundary is essential as our use case of authentication can tolerate false negatives but no false positives. We call the similarities corresponding to the data from within the local similarity and to the outside data the remote similarity. The *local similarity* describes how similar the fingerprints of a template are at least. We base the local similarity on the fingerprints of one template we know. The *remote similarity* describes how similar remote fingerprints and the template are. Ideally, we would have access to all possible fingerprints of MEs worldwide that are remote to the template in question. However, getting remote fingerprints of all other MEs is infeasible. Therefore, we approximate the remote similarity by sampling measurements from remote MEs to construct a set of remote fingerprints.

We use our decision rule to decide whether an unlabeled fingerprint is sufficiently similar to a template, based on whether the similarity of a fingerprint and a template is greater or equal to a threshold. Using the local and remote similarity, we compute a threshold per template. In Fig. 1, the threshold correlates to the boundaries around a cluster. Knowing the local similarity of a template allows us to check whether a new, unlabeled fingerprint is sufficiently similar to a template to classify as being at the same ME. Knowing the remote-similarity estimate, we can check whether a new, unlabeled fingerprint is sufficiently dissimilar to classify as being remote to the ME. Combining both, the local and the remote similarity, to the threshold of a template grants us the knowledge that at least every fingerprint observed would be classified correctly. Our decision rule is congruent with a binary classification of the fingerprint. We can phrase it as asking whether the fingerprint belongs to the same ME as the template. If so, we conclude that the ME is identical (i.e., the fingerprint was measured at the template's location). Otherwise, we classify a fingerprint as remote to the ME of the template.

3 Instantiation Using WiFi Beacon Frames

We measure WiFi signals to describe the ME. We consider the availability of the spectrum's signals and the availability of the sensor itself. WiFi is a ubiquitous signal in urban environments, with sensors being ubiquitous in consumer devices. Most WiFi access points (APs) emit a beacon frame signal to indicate their presence. A user measures these beacon frames sent by the APs in his

surroundings to conduct a measurement. Measuring the beacon frames does not require a user to connect to a WiFi AP. The user only records WiFi signals in his proximity.

We use the composition of APs to build a fingerprint of a specific ME. In the following, we refer to the WiFi receiver as *sensor* s. It measures the WiFi APs in its proximity. A *measurement* m contains a representation of the beacon-frame features received by the sensor $s(m)$. The number of features of one beacon frame depends on the AP and the sensor. The hardware capabilities determine the sensor's ability to receive beacon frames. The software determines which features of the beacon frames are accessible to the user. We denote an *access point representation (APR)* from a measurement m by $APR(m)$. We build the APR from the features provided by the AP. We distinguish the features between identifying features and capability features. The identifying features of an AP's beacon frame determine the AP uniquely. These are the service set identifier (SSID) and its media access control (MAC) address. However, both data are personally identifiable information (PII). The EU classifies a MAC address belonging to a user even in its hashed form as PII [27]. Because we ask a user to measure his environment, we can not even distinguish whether an AP belongs to the user or not. The SSID is most likely also PII as the SSID might contain names or addresses but also, in the default state, describes the vendor and model of the AP. Another problem with the SSID is that it is potentially volatile since the user can rename it. A WiFi beacon frame also contains capability features. These capability features are not PII as they encode the capabilities of the AP itself. In our case, the maximum bandwidth to use, the security and capability flags, the frequency used, and the mode of the AP. In our evaluations, we base an APR on these capability features to omit the use of PII completely.

If we use the MAC address as APR, the APR becomes unique, and a fingerprint must match a template, assuming that a set of MAC addresses only occurs at one ME (no involvement of spoofing). When using the capability features, the APRs are not unique. However, we show that using only capability features, the sets of APRs from a fingerprint are sufficiently distinguishable for different MEs. Regarding the amount of information of an APR, the MAC address [9] consists of 48 bits, the SSID of 256 bits, and the capability features are dependent on the operating system (OS). We choose the Linux OS in which the specification defines 63 bits [8]. These 63 bits are the maximum entropy possible but will likely not represent the amount of information in real-world measurements. To approximate a more realistic estimate of information per APR, we analyze our data in Sect. 5.2 to derive a lower bound for the observed information. Further, we found that privileged access on Linux (root space) and Windows allow for more capability features, while OSX and mobile operating systems seem more limited in accessing capability features.

Aside from the APR, we focus on the received signal-strength indicator (RSSI), which shows the perceived signal strength of a sensor and is denoted by $RSSI(m)$. The unit of an RSSI value is decibel, often mapped to a percentage. We normalize it such that $RSSI(m) \in [0, 1]$. The RSSI is sensor and

software dependent, which is no problem as we intend to fingerprint an ME on a per-sensor basis. We expect that including the RSSI value allows us to increase the fraction of relevant instances among the retrieved instances. We plot a measurement in Fig. 2 to illustrate the relationship of the RSSI to its proximity to the sensor while also showing the differences in the APRs.

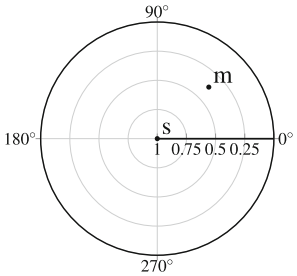


Fig. 2. A measurement m plotted in its polar coordinates. We center the sensor and have a decreasing radial coordinate $r = 1 - RSSI(m)$ being the perceived signal strength from the sensor and an angular coordinate $\varphi(APR(m)) = \frac{APR(m)}{max(APR)} * 2\pi$ being the normalized APR mapped to the unit circle.

3.1 Dataset

Online available datasets of WiFi data are either not clustered according to an ME or do not contain the beacon-frame features we focus on to build the APR. Therefore, we create our own dataset by conducting WiFi measurements, which we published.[1] We use off-the-shelf WiFi receiver hardware in laptops (e.g., [10]) as sensors and conduct our measurements with several such sensors. We keep a sensor at a fixed location for a duration of four hours. We receive $60 * 60 * 4 = 14400$ aggregated fingerprints of $t = 1$ s per ME. We consider twelve MEs recorded by ten different sensors, nine representing a home environment and three representing a work environment. We conducted all measurements at MEs such that there is no physical overlap of the received WiFi APs. Additionally, we consider a thirteenth set of fingerprints, the remote set. In total, we take 187,200 fingerprints of $t = 1$ s of the 12 MEs and the remote set into account.

3.2 Feasibility

We conduct preliminary estimates to assess whether there is a correlation between the observed APs and the ME of a sensor. To estimate the APRs' influence on the ME, we calculate the adjusted mutual information (AMI) [25]. Mutual information (MI) is the amount of information obtained about one random variable by observing another. The definition of the AMI uses H to indicate

[1] https://gitlab.com/WiFiFingerprinting/Data.

Shannon entropy [24] and the expected value E. The adjusted mutual information corrects for chance and returns a nonmetrical value. A value close to 0 indicates that there is no correlation. A value of 1 indicates that knowing feature X fully determines the label Y. The authors of [25] state that the larger the feature-to-label ratio becomes, the more the AMI approaches zero. If the ratio is larger than 100, they assume an AMI fairly close to zero. Therefore, they advise using the minimum entropy of X and Y as the normalization function if the feature-to-label ratio becomes too large. We estimate that an APR has about 10 bits (see Sect. 5.2), and we know that we have 12 MEs. We compute the AMI with X being the APRs and Y being random labels assigned to the specific MEs. We start by considering a single APR. In this case, the ratio of values of X and the number of labels Y is about 100. Therefore, we define the AMI to be:

$$AMI(X,Y) = \frac{MI(X,Y) - E\{MI(X,Y)\}}{min(H(X),H(Y)) - E\{MI(X,Y)\}}$$

We are computing the AMI per APR, which results in an AMI of 0.78. We perceive this value as too low for classification, especially for our application of authentication. To further measure the impact of APRs on the ME, we computed the AMI for tuples of APRs. Using tuples for X increases the feature-to-label ratio by several orders of magnitude, making it even more relevant to choose the minimum as the normalization function. In the 2-tuple case, we already get an AMI of 0.95. It gets closer to 1 for each increment in tuple length. For a 4-tuple, we already get an AMI of 0.98. However, the space complexity grows out of proportion. We observed between 1 and 42 single APRs per fingerprint. Therefore, we have up to $\binom{42}{k}$ k-tuples to consider.

This space complexity problem is also why classical machine learning approaches requiring one-hot encoding of categorical values require considerable space complexity during training and a significantly increased load during classification. One-hot encoding would introduce an unnecessary memory requirement on the device conducting the classification. Therefore, we use a Jaccard similarity-based approach. It does not have this space complexity problem as there is no training phase except computing the threshold once per template and our proposed mechanism only uses a list of single APRs with corresponding RSSI values as the template. We always consider the maximum available APR combinations by using the Jaccard similarity because we have seen that using more APRs increases the AMI, which is achievable from a performance point of view.

3.3 Fingerprinting the ME

We construct a fingerprint $\mathbf{F}$ of the ME in the WiFi instantiation from a set of measurements of a single sensor s during a time interval of t seconds. We expect that a fingerprint of a longer duration increased the hit rate. Additional to the sensor $s(\mathbf{F})$, and the duration $t(\mathbf{F})$, we denote the identified set of APRs as

$\mathbf{APR}(\mathbf{F})$. The number of times an AP has been measured during fingerprinting $\mathbf{F}$ we denote as:

$$|\mathbf{F}; AP| = |\{m \in \mathbf{F} | APR(m) = AP\}|$$

We denote the received signal strength of a measured AP during fingerprinting as $RSSI(\mathbf{F}; AP)$, is defined as the average of the individual measurements:

$$RSSI(\mathbf{F}; AP) = \frac{\sum_{m \in \mathbf{F}, APR(m)=AP} RSSI(m)}{|\mathbf{F}; AP|}$$

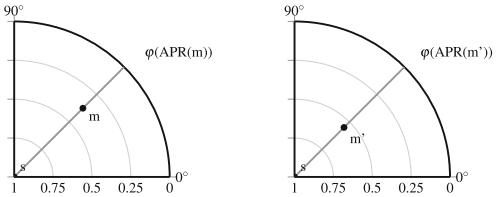
The template $\mathbb{F}$ from one ME is represented by a set of n fingerprints received by a single sensor s. Therefore, the template is constructed from $t \times n$ measurements of a single, spatially fixed sensor s. Additional to the sensor $s(\mathbb{F})$ we denote the set of APRs from the template as $\mathbf{APR}(\mathbb{F})$.

3.4 Similarity of Fingerprints and Templates

The fingerprint $\mathbf{F}$ and the template $\mathbb{F}$ are sets of APRs with their corresponding RSSI values. To compute their similarity, we apply a standard measurement for set comparison, the Jaccard similarity [12], and create a variant that considers the RSSI value. We choose the Jaccard similarity because it enables us to always consider the maximum available APs while being lightweight in terms of computational and space complexity. We denote the *Jaccard similarity* of a single fingerprint $\mathbf{F}$ and a template $\mathbb{F}$ by:

$$JS(\mathbf{F}; \mathbb{F}) = \frac{|\mathbf{APR}(\mathbf{F}) \cap \mathbf{APR}(\mathbb{F})|}{|\mathbf{APR}(\mathbf{F}) \cup \mathbf{APR}(\mathbb{F})|}$$

This version of the Jaccard similarity does not take the RSSI into account. It takes only the APR into account and computes the ratio of APRs present in the fingerprint and APRs present in the template to all APRs present in the fingerprint and the template. Using the JS implies that only the APR determines the similarity of a fingerprint and a template. Figure 3 shows a plot of this.



(a) Segment of a template $\mathbb{F}$ (b) Segment of a fingerprint $\mathbf{F}$

Fig. 3. Two segments of a fingerprint and a template. The template depicts a measurement m and the fingerprint a measurement m'. We assume that $APR(m) = APR(m')$ and that $RSSI(m) \neq RSSI(m')$. When calculating their similarity with $JS(\mathbf{F}; \mathbb{F})$, m' and m are assumed to be equal as it takes only the APR into consideration.

We construct a specific Jaccard similarity that accounts for the requirement that matching APRs have a similar RSSI value. Two RSSI values are similar if they differ at most by a difference $d(\mathbb{F})$. We denote the absolute difference, using $\|$ to denote the absolute value, of the RSSI values of an AP occurring in two fingerprints $\mathbf{F}$ and $\mathbf{F}'$ by:

$$\|\mathbf{F}; \mathbf{F}'; AP\| = \|RSSI(\mathbf{F}'; AP) - RSSI(\mathbf{F}; AP)\|$$

The similarity measure taking the RSSI into account works on a subset of $\mathbb{F}$. This subset is defined such that the RSSI values of a measurement with a matching APR from a fingerprint $\mathbf{F}$ and a template $\mathbb{F}$ differ at most by $d(\mathbb{F})$, for $d(\mathbb{F}) \in [0, 1]$. If $d(\mathbb{F}) = 0$, the RSSI values of a measurement in a fingerprint $\mathbf{F}$ and a template $\mathbb{F}$ have to be equal. The size of $d(\mathbb{F})$ determines how many RSSI values are accepted to be sufficiently similar. Thus, the closer $d(\mathbb{F})$ becomes to 1, the more likely it is that the similarity taking the RSSI value into account is equal to its counterpart that does not take the RSSI value into account. We compute $d(\mathbb{F})$ per template and define it as:

$$d(\mathbb{F}) = max\{\|\mathbf{F}; \mathbf{F}'; AP\| \mid AP \in \mathbf{APR}(\mathbf{F}) \cap \mathbf{APR}(\mathbf{F}'), \mathbf{F}, \mathbf{F}' \in \mathbb{F}, \mathbf{F} \neq \mathbf{F}'\}$$

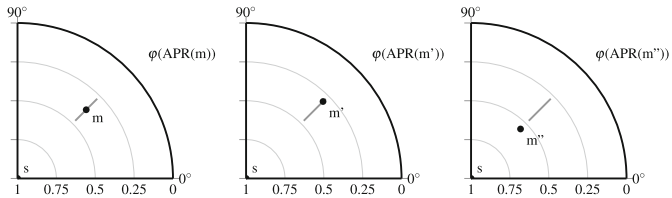
Choosing $d(\mathbb{F})$ to be a model parameter allows us to cover the difference of the RSSI values measured from the same APR observed in a fingerprint $\mathbf{F}$ and in a template $\mathbb{F}$. We define the subset as:

$$\{\mathbf{F}; \mathbb{F}\} = \{\mathbf{F}' \in \mathbb{F} \mid (\|\mathbf{F}; \mathbf{F}'; AP\| < d(\mathbb{F})) \text{ with } AP \in \mathbf{APR}(\mathbf{F}) \cap \mathbf{APR}(\mathbf{F}')\}$$

Assuming this subset, we can construct the similarity measure, which considers the RSSI value. The *Jaccard similarity with RSSI* of a single fingerprint $\mathbf{F}$ and a template $\mathbb{F}$ is defined as:

$$JSR(\mathbf{F}; \mathbb{F}) = \frac{|\mathbf{APR}(\{\mathbf{F}; \mathbb{F}\})|}{|\mathbf{APR}(\mathbf{F}) \cup \mathbf{APR}(\mathbb{F})|}$$

Whether a measurement with $APR(m)$ in a fingerprint $\mathbf{F}$ and in a template $\mathbb{F}$ is assumed to be the same is dependent on the RSSI value. A measurement m must fulfill two requirements to be an element of the intersection. First, the $APR(m)$ must occur in the fingerprint and the template. Second, the $RSSI(m)$ from the fingerprint and the template must differ at most by the fixed distance $d(\mathbb{F})$. Figure 4 depicts a plot of this.



(a) Segment of a template $\mathbb{F}$ (b) Segment of a fingerprint $\mathbf{F}$ (c) Segment of a fingerprint $\mathbf{F}'$

Fig. 4. Three segments of a template ($\mathbb{F}$) with a measurement m and its RSSI difference $d(\mathbb{F}) = 0.1$, and of the fingerprints $\mathbf{F}$ and $\mathbf{F}'$. In a) the measurement is depicted by a dot labeled m and the difference $d(\mathbb{F})$ is depicted by a line with a length of 2d (added and subtracted from the $RSSI(m)$ of the measurement in the template). In segment b) the measurement is depicted by m' such that $APR(m') = APR(m)$. It shows an RSSI value inside the boundary (i.e. $RSSI(m') > RSSI(m) - 0.1$). In segment c) the measurement is depicted by m'' such that $APR(m'') = APR(m)$. It shows an RSSI value outside the boundary (i.e. $RSSI(m'') > RSSI(m) + 0.1$).

3.5 Determining the Similarity Threshold

We compute the threshold per template and per similarity measurement $SIM \in \{JS, JSR\}$. The local similarity is the lowest similarity observed between each fingerprint that makes up a particular template and the template itself. This guarantees that all fingerprints used to build the template are more similar than the threshold and implies that a classifier based on this threshold would classify all fingerprints used to build the template correctly. We define it as:

$$local(SIM; \mathbb{F}) = min(\{SIM(\mathbf{F}, \mathbb{F}) \mid \mathbf{F} \in \mathbb{F}\})$$

We also use fingerprints from remote MEs as a reference to determine the threshold. Such a fingerprint $\mathbf{R}$ is built from measurements that are not at the ME of a template. We build a diverse set of fingerprints from these remote fingerprints $\mathbb{R}$. We cap the number of APRs per fingerprint $\mathbf{R}$ at the maximum number of observed APRs in the template to allow for a similarity-based comparison. The remote similarity is the largest similarity observed between each remote fingerprint taken into account and the template itself. This guarantees that all remote fingerprints from $\mathbb{R}$ have a similarity lower than the threshold. Implying that a classifier based on this threshold would classify all remote fingerprints correctly (not belonging to $\mathbb{F}$). We define it as:

$$remote(SIM; \mathbb{F}; \mathbb{R}) = max(\{SIM(\mathbf{R}, \mathbb{F}) \mid \mathbf{R} \in \mathbb{R}\})$$

We compute the **threshold** T by taking the average of the local and the remote similarity, using both balances the threshold by introducing some tolerance. We expect a new fingerprint at the ME with a slightly lower similarity than observed to have a higher similarity than the threshold. We also expect a new remote fingerprint with a slightly higher similarity than the similarities observed to have a lower similarity than the threshold. Choosing the threshold to be the local and remote

similarity average prevents the overfitting of the training data used to create the threshold. Given a similarity measure SIM, a template $\mathbb{F}$ and a set of remote fingerprints $\mathbb{R}$ we denote the threshold by:

$$T(SIM; \mathbb{F}; \mathbb{R}) = \frac{(local(SIM; \mathbb{F}), remote(SIM; \mathbb{F}; \mathbb{R}))}{2}$$

3.6 Classifying Fingerprints

The necessary elements to define our decision rule as a classifier are the components and tools for comparison we defined. Our components are: a measurement, the fingerprint, and the template. Our tools for comparison are: the similarities and a template-specific threshold. Our fingerprinting mechanism aims to classify a fingerprint $\mathbf{F}$ by computing its similarity with a template $\mathbb{F}$. If the similarity is larger than the template-specific threshold, we classify the fingerprint as being at the template's ME. Otherwise, we classify the fingerprint as remote to the template's ME. The classifier returns this Boolean decision and we denote it by:

$$C(SIM; \mathbf{F}; \mathbb{F}; \mathbb{R}) = SIM(\mathbf{F}; \mathbb{F}) > T(SIM; \mathbb{F}; \mathbb{R})$$

The classifier C allows us to fingerprint the ME in terms of ubiquitous WiFi beacon frames. It is limited only by the surrounding WiFi APs and the time $t(\mathbf{F})$ required to fingerprint a ME.

4 Performance

We distinguish two models for classification based on the similarity measure used. We either only compare the APRs by applying the JS, or we also consider the RSSI values and apply the JSR. We also vary the time t (in seconds) listened in for a WiFi fingerprint. We started our experiments with $t = 20$ s and increased it incrementally to one minute. Further, we vary the number of WiFi fingerprints n to build a template. We take n to represent a template of the equivalent of 15 min, 30 min, one hour, and two hours (e.g., assuming $t = 20$, the two hours of fingerprint data accumulate to $n = 360$ fingerprints used to build a template). The model parameters are the threshold $T(SIM, \mathbb{F}, \mathbb{R})$ and the tolerated RSSI difference $d(\mathbb{F})$. We compute the RSSI distance $d(\mathbb{F})$ from a template. In our data set, we observe only a slight variation for the values for $d(\mathbb{F})$ per template (at most 0.08), which confirms our expectation that the RSSI increases the uniqueness of measurements.

4.1 Classification

We classify a fingerprint $\mathbf{F}$ based on a template $\mathbb{F}$ and a threshold $T(SIM, \mathbb{F}, \mathbb{R})$. We assume the template to be known for the classification step and derive the threshold from the fingerprints used for building the template and the set of remote fingerprints $\mathbb{R}$. Per ME, we consider fingerprints taken for the total

duration of four hours ($t = 14400$). We split the observed fingerprints in half. The latter half, the second two hours, always form the test set. For the training set, we vary the number of fingerprints used in the first half to represent 15 min, 30 min, one hour, and two hours. We compute the similarity of the fingerprint and the template. If the fingerprint's similarity to the template is larger or equal to the threshold, we consider the fingerprint to be measured at the same ME as the template.

To determine the performance of our classifier, we look at its precision and recall. Both are defined in terms of the predicted conditions. The *precision* measures the true positives among all positives predicted. It is defined as: $precision = \frac{TP}{TP+FP}$. The focus is on the false positives. In our context, these translate to remote fingerprints, which classify as being at the ME. The *recall* measures the true positives among all real positives. It is defined as: $recall = \frac{TP}{TP+FN}$. The focus is on the false negatives. In our context, these translate to fingerprints of the ME, which we classify as not being at the ME. The *accuracy* measures correct predictions among the total number of cases examined. It is defined as: $accuracy = \frac{TP+TN}{TP+TN+FP+FN}$. It combines the false positives and the false negatives in one value.

4.2 Validation

We partition our data into two sets: fingerprints of the ME and remote fingerprints. The fingerprints of the ME form the template and the input for computing the local similarity per ME. The remote fingerprints form the input for computing the remote similarity, a limited collage of WiFi fingerprints of various MEs. Both directly influence our threshold T.

We deal with a limited set of remote fingerprints, with different fingerprints from different MEs. To verify the impact of different compositions of fingerprints used to construct the remote similarity, we conduct a Monte Carlo cross-validation (MCCV). Per ME, we randomly shuffle the remote fingerprints before splitting them between the training and the test set. We chose MCCV due to our limited remote fingerprints' set size and our classification mechanism. We deal with an equally partitioned set. Hence, folded cross-validation would not provide a sufficient answer. Even though we deal with a time series, rolling cross-validation is not required for the remote fingerprints. We apply the MCCV v times and choose $v = n$, the training and test sets' sizes. The MCCV allows us to test for v compositions of the remote fingerprint data we consider. We present the precision and recall as our results per MCCV iteration.

4.3 Results

We conduct classifications for all MEs' fingerprints according to the structure described in Sect. 4.1. We start with taking fingerprints of length $t = 20$ s into account and compute a template from $n \in \{45, 90, 180, 360\}$ fingerprints. Our results show that building a template from more fingerprints, a larger n, provides more information about the ME, which is consistent with our expectations.

However, we are aiming for higher precision and recall. The Jaccard similarity taking the RSSI into account produces a much more stable result. We continued with taking fingerprints of length $t = 30$ s into account and compute a template from $n \in \{30, 60, 120, 240\}$ fingerprints. Again the Jaccard similarity taking the RSSI into account produces a much more stable classification result. Still, we are aiming for higher precision and recall. However, we observe a smoothing compared to the $t = 20$ s case. This trend continues and provides near-perfect results when we take fingerprints of length $t = 60$ s into account. We compute a template from $n \in \{15, 30, 60, 120\}$ fingerprints. All choices of n provide promising results. The precision in the case of the JS is lower when most of the time, only a single AP can be measured (e.g., the ME labeled $L8$). For the JS and the JSR, we show the precision per MCCV iteration in Fig. 5. In the case of the JSR, all MEs have a precision greater than 0.98.

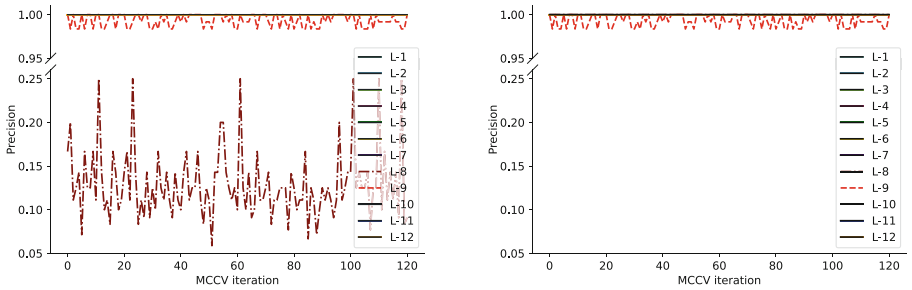


Fig. 5. Precision for the case of JS (left) and JSR (right) with $t = 60$ and $n = 120$, for $v = 120$ MCCV iterations.

The recall in the case of the JS is for all MEs greater than 0.99, except $L8$. It has a recall very close to zero. It would never classify the location $L8$ correctly because it mainly contains only a single AP. For the JSR, all other MEs have a recall greater than 0.99. On the one hand, the recall of ME $L8$ becomes 0.975. On the other hand, the recall of the ME $L5$ has a larger variation. We show the JS's and JSR's recall per MCCV iteration in Fig. 6.

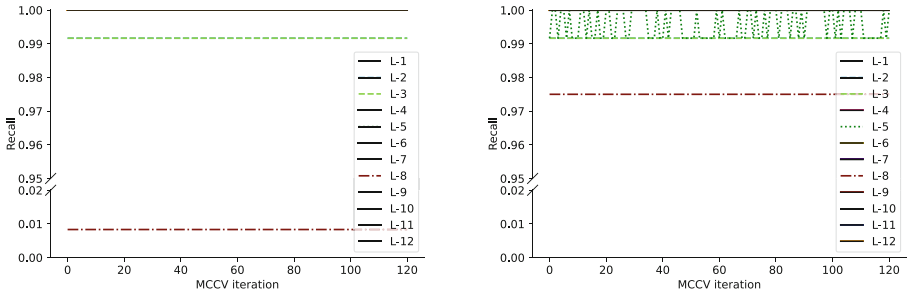


Fig. 6. Recall for the case of JS (left) and JSR (right) with $t = 60$ and $n = 120$, for 120 MCCV iterations.

We observed that choosing a larger n, thus using more fingerprints to build a template, lowers the number of false negatives, which is consistent with our expectations. If we have a more detailed picture of the situation, we can map a new observation more accurately. The consistent performance of the JSR in contrast to the JS confirms our expectations that similar RSSI values allow us to map a newly observed AP with higher precision. By performing better than our AMI estimate (0.98 with only four APs) we also confirm our expectation as we consider all APs. The ME labeled $L8$ even shows that MEs with too few (e.g., one) APRs should not be used.

5 Use Case: Nonintrusive Authentication

We now propose to use the ME as a nonintrusive authentication factor. In this setting, the fingerprint of an ME becomes the authentication claim validated against a template. We focus on extending an existing two-factor authentication scheme. Therefore, the presence of a first authentication factor (e.g., the password) and the presence of a second authentication factor (e.g., SMS token) are a given. To conduct authentication, we distinguish between a user and a service. We assume the user to have a WiFi-capable sensor with software capable of conducting measurements and the service to be capable of registering and authenticating a user. We distinguish three phases of authentication:

Registration: The service and the user agree upon t, the number of seconds listened in to build a fingerprint, and n, the number of fingerprints used to build a template. In the following, the user conducts $n * t$ stationary measurements and combines them to a template $\mathbb{F}$. This template is stored on the service side after successfully authenticating with the previously mentioned first and second factors. The server is further assumed to hold a remote set $\mathbb{R}$.

Login: The user's sensor conducts t stationary measurements and combines them into a fingerprint $\mathbf{F}$. The user then sends this fingerprint $\mathbf{F}$ to the service after being authenticated with the mentioned first and second factors.

Verification: The service has a template of a registered user. To verify the authentication request of a user, the service obtains the fingerprint $\mathbf{F}$ of the login phase. It classifies whether the obtained fingerprint is sufficiently similar to the user's template by computing the threshold and decision rule specified in Sect. 2.3.

5.1 Augmenting Existing Schemes

We present a way to augment any authentication scheme with our ME. The service can invoke the second factor (e.g., a software token) either in parallel or only if the fingerprint is not sufficiently similar. If used in parallel, the ME becomes a third authentication factor granting nonintrusiveness and consistency of the environment. If the second factor is used only in doubt, the ME becomes the de facto second factor. It also grants nonintrusiveness and the consistency of

the environment but in addition lifts the user's burden to engage in the second authentication factor actively. The service can request a fingerprint at any time. Only the fingerprint's length, t seconds, limits the interval to conduct authentication. Our results suggest that the service can attempt authentication at least every minute (see Sect. 4.3).

5.2 Factor Strength

To determine the strength of our authentication factor, we compute the amount of information in terms of Shannon entropy H [24]. The entropy in our scheme resides in the APRs of a fingerprint. Each APR has a theoretical upper bound of 63 bits according to the specifications [8]. However, the measurable entropy of APRs will only be a fraction of these possibilities. Therefore, we calculate a minimum estimate based on the APRs of the 187,200 fingerprints observed in our data set. We get a minimum entropy of $H(MAC) = 10.45$ bits considering the observed MAC addresses. We take the joint entropy of dependent variables into account for the capability features before summing them together. This results in $H(Flags, WPA, RSN) + H(Frequency, Bitrate) + H(Mode) = 9.1$ bits of minimum entropy per APR expressed in terms of its capability features. To estimate a lower bound for the entropy provided in one fingerprint, we need to consider the number of observed APs. In our data set, each fingerprint contains one to forty-two APs. Therefore, each fingerprint $\mathbf{F}$ provides at least $9.1 \times |\mathbf{APR}(\mathbf{F})|$ bits of entropy when using the capability features.

5.3 Adversary Model and Security Analysis

Masquerading is the main threat against any authentication scheme. For fingerprinting MEs this implies to get data that the service accepts as a valid fingerprint. We look specifically at attacks of acquiring a fingerprint. We consider the compromise of the user devices as an orthogonal problem and assume that known device protection techniques are in place, such as the regular installation of security updates. Furthermore, we assume that a public-key infrastructure (PKI) is in place to guarantee that the service is eligible to query the user's ME and that the communication channel is TLS [6] protected.

Brute Force. An adversary can try to guess the fingerprint of the user. The recommendations of the NIST has varying factor-strengths based on the factors themselves; a user-chosen password requires a minimum of 48 bits, a key for an attestation of a sensor-modality in the biometric context a minimum of 112 bits [19]. Fingerprinting the ME resides somewhere between. It is not arbitrarily chosen by the user like a password, but can and will change unlike biometric features. Assuming a minimum of six APs, an adversary has to brute force at least 2^{54} possibilities, which is larger than the entropy recommendations for passwords. Even if the adversary can guess a fingerprint, the first and second authentication factors are still in place.

Compromise Template Confidentiality. An adversary can try to compromise the user's template by taking control of the service. What can be done is not to store the user's template $\mathbb{F}$ in plaintext. The only information required in plain text is the RSSI value to compare its distance to a newly obtained RSSI value. However, the RSSI value is not identifying. Storing a salted and hashed representation of each APR, congruent to the password case, is insufficient. The entropy of an individual APR is too low. Knowing the salt, the adversary could brute force the hash. An alternative is the salted-challenge-response authentication mechanism (SCRAM). The crucial point is that SCRAM applies a password-based key derivation function (PBKDF2), which uses a client-side salt to increase the entropy. Using SCRAM solves the low entropy problem of a single APR. To not further burden the user by requiring him to store a salt for each observed APR, we assume the user will use his first AF password as salt input to each PBKDF2. Note that this does not violate key separation as the reused password is used as salt to the PBKDF2, while each observed APR is used as the 'password'. The guarantees provided by SCRAM can then be transferred to a set of APRs.

Compromise the Communication Channel. An adversary can also try to attack the communication channel. In a successful attack, the confidentiality of the exchanged information would be compromised. In the registration phase, this exchanged information contains the user's template; during the login phase, it contains a fingerprint. First, the assumption of a TLS-protected communication channel comes into mind. However, even if the TLS is broken or circumvented, applying SCRAM stops an adversary from attacking the communication channel. SCRAM exchanges several hashed messages between the user and the service that are not susceptible to replay attacks. It authenticates the client to the server and the server to the client.

On the Intrinsic Threat of Local Adversaries. Using location as an authentication factor has the intrinsic property that everyone at the location classifies as being at the location. Therefore, we assume that our authentication factor is only used in conjunction with existing authentication factors. The authors of [3] claim to define a location with an accuracy of 2 m based on the RSSI difference $d(\mathbb{F})$ when combining WiFi with Bluetooth data. We intend to investigate the boundaries of an ME by WiFi with a suitable dataset in the future. However, being on location is limited to a very restricted set of adversaries who also need to acquire the other authentication factors.

Too Few APs in the ME. If a template contains too few (less than six) APs, the service should advise against using the ME as an authentication factor because the entropy is too low. Our experiments show that our classification works for locations with only a single AP. However, the security guarantees are insufficient. Our results show with a precision of 1 that we do not grant authentication in a situation in which authentication should have been denied.

6 Comparison with Related Work

Continuous authentication systems in the literature are based on the physical activities [1] or biometric features of the user [17]. Behavioral biometrics also include the routines of a user. Several works consider building such routine profiles of a user's day from WiFi. The authors of [20] restrict their profile creation to the SSID of only the AP to which the user is connected. They also use several other sensor readings. The authors of [14] apply a similar approach to large available datasets. The authors of [18] compare application usage, Bluetooth, and WiFi signals. They report stable rates above 90% for WiFi over one week using the combination of MAC, SSID, and RSSI, showing that our expectation of the consistency of WiFi APs is valid. In Table 1, we compare our results with the related literature (reporting the accuracy). This comparison shows two significant differences compared to our approach. First, all works from the literature use more sensors than just WiFi sensors. Second, all works using WiFi sensors use the identifying features (MAC address or SSID), which are unique, but privacy sensitive. If at least six APs are available, our mechanism outperforms the results reported in the literature with an accuracy of 0.996. We achieve this while only taking capability features and no PII into account. All other works build upon potentially privacy-invasive data. One explanation might be that we do not focus on creating a user's behavioral profile. Doing so is just one application of using the ME as an authentication factor. Also, setting a minimum of observed APs explains our slight edge in performance. The closest result reported [20] takes only a single connected AP into account, and the second closest [3] focuses only on the top-ranked network nodes (determined by the RSSI value).

Table 1. Comparison of our mechanism with results from the literature in location fingerprinting in terms of the signals used, the type of features used, and the reported accuracy.

Study	Signal	Feature	Accuracy
Ours ($\#AP \geq 6$)	WiFi	Capability	0.996
[20]	WiFi, BT, GPS, usage	Identifying	0.994
[3]	WiFi, BT	Identifying	0.984
Ours	WiFi	Capability	0.984
[14]	WiFi, App, various	Identifying	0.983
[18]	WiFi, BT, App	Identifying	0.85

Several works in the literature propose to use APs and their beacon frames to authenticate a user's location. Despite pretty good results, all of these works require a change of the beacon frame itself. Cho et al. [5] propose a protocol to allow location-aware access control by defining a location area enclosed by overlapping ranges of multiple APs. They derive a location key from the overlapping APs' beacon frame information but require the APs to include a nonce

into their beacon frames and communicate them via a secured channel among themselves. Bao [4] proposes a solution assuming an additional key server next to the user and APs. Saroiu et al. [23] go even one step further and suggest adding two functionalities to the AP. Their AP must be capable of generating a location certificate and engaging in an exchange protocol on the user's behalf. Pham et al. [21] observed this issue and suggested reducing its surface by relying on a centrally coordinated distributed AP infrastructure by proposing to rely on existing WiFi APs operated as hotspots owned by the WiFi network providing company. Our solution differs from these, as we do not require the AP to send content in the beacon frame that exceeds the beacon frame standard [9].

7 Discussion and Future Work

Changes in the ME. Signal availability is a problem for all radio communications. However, the results of [18] suggest consistency of at least one week for WiFi signals, and we expect that APs are far less frequently changed. We believe that an AP is more likely to be exchanged by the user switching ISPs than reconfiguration, which occurs less than annually per AP. Even if some APs change, our classifier construction allows for tolerance in the composition of signals. If the ME changes too much, the user should update the template by re-enrolling an ME. The first and second authentication factors provide the authenticity of the user in this scenario.

Keeping the APRs Confidential. A fruitful future work might be to introduce a user-specific secret S and not store the APRs in plain text but a cipher text after encryption under S. Using a user-specific secret S has the downside that this secret must be securely stored on the user's device. However, it would serve at least three purposes: Firstly, MAC addresses and other PII may be used because they are unretrievable from a cipher text without knowing the secret S but will most likely improve the classifier's performance. Secondly, the problem of local adversaries being capable of plainly conducting a measurement of the user's location becomes impossible as such an adversary would be required to get hold of S. Thirdly, a chain of trust links a measured environment to a user. The cipher text would be constructed from the measurement of the ME and the user-specific secret S. This binds the measurement to the secret S. Assuming that S is stored securely on the device, the cipher text is bound to the device. If the device is bound to the user either by knowledge or biometry, there is a chain of trust, from a measurement to a cipher text, from the cipher text to a device, and from the device to the user.

Behavioral user Profiles. Another direction to look into is the user's behavior. When a user consistently uses a service by authenticating via the ME, it could become possible to build a behavioral profile (e.g., between 09.00 and 10.00 AM, the user recurringly logs in from the same ME during workdays). Creating a behavioral profile is a double-edged sword. On the one hand, it could improve the precision of an authentication claim, and the system could learn those behavioral

patterns to increase the likelihood of authenticating the correct user. On the other hand, there might be contexts where the service should not learn about a user's template correlation. A future direction to look into is building these profiles or preventing the service from knowing which template a user's claim matches.

Further Directions. We envision further research in evaluating the perimeters of MEs to understand how close a fingerprint must be to an ME to match a template and, thus, how close an adversary must be to gather a matching fingerprint. Moreover, we see interesting future work in using the entropy in an ME's fingerprint to harden an existing authentication factor.

8 Conclusion

We introduce a nonintrusive authentication factor based on the user's measurable environment (ME). We aim for simplicity and consistency while respecting the privacy of the user. Simplicity is provided by lifting the burden for the user to perform extra tasks for multi-factor authentication (e.g., retype SMS tokens on every login). We respect the user's privacy by conducting this authentication in our instantiation only based on capability features and by only recognizing a known ME instead of mapping the user to a geographical location.

We show that a WiFi fingerprint (not containing PII) can be classified with a precision of 1 and a recall above 0.99 when observing multiple APs. A precision of 1, no false positives, enables us to apply the concept of WiFi fingerprinting to authentication since no wrongful authentication gets conducted. A recall above 0.99 implies that our mechanism correctly authenticates a valid user most of the time, which is tolerable as we assume a first and second factor as a backup for authentication. We rely on APs that are ubiquitously present in any WiFi environment. We require only a few kilobytes of data to be transmitted, and the classification requires only a low amount of complexity. We perform the classification of an ME by computing the JSR of a newly obtained fingerprint to a known template and using it as a nonintrusive authentication factor.

References

1. Abuhamad, M., Abuhmed, T., Mohaisen, D., Nyang, D.: AUToSen: deep-learning-based implicit continuous authentication using smartphone sensors. IEEE Internet Things J. **7**(6), 5008–5020 (2020)
2. Al-Naji, F.H., Zagrouba, R.: A survey on continuous authentication methods in internet of things environment. Comput. Commun. **163**, 109–133 (2020)
3. Alawami, M.A., Kim, H.: LocAuth: a fine-grained indoor location-based authentication system using wireless networks characteristics. Comput. Secur. **89**, 101683 (2020)
4. Bao, L.: Location authentication methods for wireless network access control. In: 2008 IEEE International Performance, Computing and Communications Conference, pp. 160–167 (2008)

5. Cho, Y., Bao, L., Goodrich, M.T.: LAAC: a location-aware access control protocol. In: 2006 3rd Annual International Conference on Mobile and Ubiquitous Systems-Workshops, pp. 1–7 (2006)
6. Dierks, T.: TLS v 1.2 (2008). http://www.hjp.at/doc/rfc/rfc5246.html
7. Gassend, B., Clarke, D., Van Dijk, M., Devadas, S.: Silicon physical random functions. In: Proceedings of the 9th ACM Conference on Computer and Communications Security, pp. 148–160 (2002)
8. GNOME. org.freedesktop.networkmanager.accesspoint (2021). https://developer.gnome.org/NetworkManager/1.2/gdbus-org.freedesktop.NetworkManager.AccessPoint.html
9. IEEE Standard. Wireless LAN medium access control (MAC) and physical layer (PHY) specifications (2007). https://www.iith.ac.in/tbr/teaching/docs/802.11-2007.pdf
10. Intel. Dual band wireless-ac 8265 (2021). https://ark.intel.com/content/www/us/en/ark/products/94150/intel-dual-band-wireless-ac-8265.html
11. ISO 27000. Information technology, security techniques, information security management systems, overview andvocabulary (2018)
12. Jaccard, P.: Étude comparative de la distribution florale dans une portion des alpes et des jura. Bull. Soc. Vaudoise Sci. Nat. **37**, 547–579 (1901)
13. Jeong, W., et al.: SDR receiver using commodity WiFi via physical-layer signal reconstruction. In: Proceedings of the 26th Annual International Conference on Mobile Computing and Networking, pp. 1–14 (2020)
14. Kayacik, H.G., Just, M., Baillie, L., Aspinall, D., Micallef, N.: Data Driven Authentication: On the effectiveness of user behaviour modelling with mobile device sensors (2014)
15. Lebovic, N.: Biometrics, or the power of the radical center. Crit. Inq. **41**(4), 841–868 (2015)
16. McKenna, S.J., Gong, S.: Non-intrusive person authentication for access control by visual tracking and face recognition. In: International Conference on Audio-and Video-Based Biometric Person Authentication, pp. 177–183 (1997)
17. Mosenia, A., Sur-Kolay, S., Raghunathan, A., Jha, N.K.: CABA: continuous authentication based on BioAura. IEEE Trans. Comput. **66**(5), 759–772 (2017)
18. Neal, T.J., Woodard, D.L., Striegel, A.D.: Mobile device application, bluetooth, and Wi-Fi usage data as behavioral biometric traits. In: 2015 IEEE 7th International Conference on Biometrics Theory, Applications and Systems (BTAS), pp. 1–6 (2015)
19. NIST. Digital identity guidelines, authentication and lifecycle management (2021). https://pages.nist.gov/800-63-3/sp800-63b.html
20. Pang, X., Yang, L., Liu, M., Ma, J.: Mineauth: mining behavioural habits for continuous authentication on a smartphone. In: Australasian Conference on Information Security and Privacy, pp. 533–551 (2019)
21. Pham, A., Huguenin, K., Bilogrevic, I., Dacosta, I., Hubaux, J.-P.: SecureRun: cheat-proof and private summaries for location-based activities. IEEE Trans. Mob. Comput. **15**, 08 (2016)
22. Rudd, E.M., Boult, T.E.: Caliper: continuous authentication layered with integrated PKI encoding recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 127–135 (2016)
23. Saroiu, S., Wolman, A.: Enabling new mobile applications with location proofs. In: Proceedings of the 10th Workshop on Mobile Computing Systems and Applications, pp. 1–6 (2009)

24. Shannon, C.E.: Prediction and entropy of printed English. Bell Syst. Tech. J. **30**, 50–64 (1951)
25. Vinh, N.X., Epps, J., Bailey, J.: Information theoretic measures for clusterings comparison: variants, properties, normalization and correction for chance. J. Mach. Learn. Res. **11**, 2837–2854 (2010)
26. Wen, M., Hanwen, L.: Radar detection for 802.11 a systems in 5 GHz band. In: Proceedings of 2005 International Conference on Wireless Communications, Networking and Mobile Computing, vol. 1, pp. 512–514. IEEE (2005)
27. WP29. Opinion 01/2017 on the proposed regulation for the eprivacy regulation (2002/58/EC) (2017)