Perils of Zero-Interaction Security in the Internet of Things

MIKHAIL FOMICHEV*, TU Darmstadt, Germany
MAX MAASS*, TU Darmstadt, Germany
LARS ALMON, TU Darmstadt, Germany
ALEJANDRO MOLINA, TU Darmstadt, Germany
MATTHIAS HOLLICK, TU Darmstadt, Germany

The Internet of Things (IoT) demands authentication systems which can provide both security and usability. Recent research utilizes the rich sensing capabilities of smart devices to build security schemes operating without human interaction, such as zero-interaction pairing (ZIP) and zero-interaction authentication (ZIA). Prior work proposed a number of ZIP and ZIA schemes and reported promising results. However, those schemes were often evaluated under conditions which do not reflect realistic IoT scenarios. In addition, drawing any comparison among the existing schemes is impossible due to the lack of a common public dataset and unavailability of scheme implementations.

In this paper, we address these challenges by conducting the first large-scale comparative study of ZIP and ZIA schemes, carried out under realistic conditions. We collect and release the most comprehensive dataset in the domain to date, containing over 4250 hours of audio recordings and 1 billion sensor readings from three different scenarios, and evaluate five state-of-the-art schemes based on these data. Our study reveals that the effectiveness of the existing proposals is highly dependent on the scenario they are used in. In particular, we show that these schemes are subject to error rates between 0.6% and 52.8%.

CCS Concepts: • **Security and privacy → Authentication**; *Multi-factor authentication*;

Additional Key Words and Phrases: Context-based Security, Secure Device Pairing, Authentication, Internet-of-Things

ACM Reference Format:
Mikhail Fomichev, Max Maass, Lars Almon, Alejandro Molina, and Matthias Hollick. 2019. Perils of Zero-Interaction Security in the Internet of Things. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 1, Article 10 (March 2019), 38 pages. https://doi.org/10.1145/3314397

1 INTRODUCTION

The Internet of Things (IoT) has become a reality, with many smart devices communicating autonomously to provide users with seamless services like climate control at home, security systems in the workspace, and traffic management in public areas. To provide those services, IoT devices use multiple sensors to "perceive" their environment and act on it to make appropriate decisions.

Numerous security incidents [16] have shown that establishing and maintaining secure communications between IoT devices is challenging. First, centralized approaches such as Public Key Infrastructures (PKIs) are impractical due to their high complexity and limited scalability [5]. Second, many IoT devices do not feature user interfaces, making it impossible to perform security establishment using traditional mechanisms like passwords [7]. To address this problem, recent research proposed using device sensor readings of the ambient environment,

*Both authors contributed equally to this work. This is a preprint of the paper, released under rights of use according to the German UrhG.

Authors' addresses: Mikhail Fomichev, Secure Mobile Networking Lab, TU Darmstadt, Germany, mfomichev@seemoo.tu-darmstadt.de; Max Maass, Secure Mobile Networking Lab, TU Darmstadt, Germany, mmaass@seemoo.tu-darmstadt.de; Lars Almon, Secure Mobile Networking Lab, TU Darmstadt, Germany, lalmon@seemoo.tu-darmstadt.de; Alejandro Molina, Machine Learning Group, TU Darmstadt, Germany, molina@cs.tu-darmstadt.de; Matthias Hollick, Secure Mobile Networking Lab, TU Darmstadt, Germany, mhollick@seemoo.tu-darmstadt.de.

often called *context information* [22]. This information is used to build context-based security schemes operating without user interaction such as zero-interaction pairing (ZIP) [20, 24, 39] and zero-interaction authentication (ZIA) [14, 28, 36]. We further refer to both as zero-interaction security (ZIS) schemes.

The security of ZIS schemes is based on the assumption that context information has high entropy, changes frequently, and is unpredictable from outside the specified environment [31]. Context information, obtained from the ambient environment of an IoT device, is used to derive a shared secret key between colocated devices in ZIP or to serve as a proof of physical proximity between devices in ZIA. For example, similarity in ambient audio sensed by two colocated devices was successfully used in both ZIP [24] and ZIA [14], with the latter scheme becoming part of a commercial product [11]. Other research explored the applicability of different context information in ZIS schemes: temperature, humidity, pressure, and luminosity [20, 28], magnetic fields, acceleration and rotation rates [23, 26], as well as observed WiFi and Bluetooth beacons [36].

ZIS schemes have three main advantages compared to traditional approaches. First, they offer high usability by minimizing user involvement in pairing and authentication procedures. Second, ZIS schemes can scale to a large number of devices, including those that do not share a common sensing modality [13]. Third, ZIS schemes can be built on top of devices' sensing capabilities, reducing modification overhead and facilitating interoperability.

Despite the great potential of ZIS schemes to enable a more secure and usable IoT, prior work raised questions about their practical applicability [25] and security soundness [29, 30]. The evaluation of the proposed schemes in realistic IoT scenarios is crucial yet mostly missing. In our work, we fill this gap by conducting the first large-scale comparative study of existing ZIS schemes. We reproduce five state-of-the-art ZIS schemes [14, 20, 24, 28, 36] and evaluate their ability to distinguish authorized and unauthorized devices on comprehensive datasets of context information collected in three realistic IoT scenarios: a *connected car*, *smart office* and *smart office with mobile heterogeneous devices*. Our evaluation reveals trade-offs between different kinds of context information and context features, and gives insights into pitfalls of the reproduced schemes in practice.

In our scenarios, we collect seven different kinds of context information, given in Table 1: audio, WiFi and Bluetooth Low Energy (BLE) beacons, barometric pressure, humidity, luminosity, and temperature, from which we compute 16 distinct context features. We implement IoT scenarios by distributing sensing devices among various spots, each reflecting a potential IoT functionality like a smart light, with people following their daily routine in these scenarios. When evaluating mobility, we additionally supply users with different sensing devices.

From our car, office and mobile scenarios, we collect context information datasets of 1.7, 214 and 23.2 GB, respectively, and annotate them with the ground truth. To our knowledge, these are the largest datasets of annotated context information collected in the ZIS domain so far. Our analysis reveals that many of the reproduced schemes are challenged by our scenarios and often cannot maintain the classification accuracy found by their original authors, reaching error rates between 0.6% and 52.8%. We also observe that many schemes have limited adaptability to difficult circumstances and frequently are not robust, with parameters optimal for one scenario leading to notably lower classification performance in the other.

To facilitate future research, we publicly release the collected context information in an anonymized form, ground truth information, the computed context features, machine learning datasets and results for all reproduced schemes, as well as the full source code used in data collection and evaluation procedures, including metadata for reproducibility (cf. Section 3.5) [9]. We further enhance reproducibility by releasing raw audio recordings from the mobile scenario [8], making our dataset the first of its kind in the domain of ZIS.

In summary, we make the following contributions:

- We reproduce five state-of-the-art ZIS schemes [14, 20, 24, 28, 36] and design three realistic IoT scenarios from which we collect comprehensive datasets of diverse context information.
- We evaluate the scheme performance and robustness for use in different scenarios. We also provide insights into pitfalls of the reproduced schemes.

- We release the first open-source toolkit, containing datasets of diverse context information, including audio, together with the source code used to collect these data and implementations of the five ZIS schemes.

Table 1. Context information use by the reproduced schemes

ZIS scheme	Audio	BLE	WiFi	Press.	Hum.	Lum.	Temp.	Details
Karapanos *et al.* [14]	✓	–	–	–	–	–	–	§A.1
Schürmann and Sigg [24]	✓	–	–	–	–	–	–	§A.2
Miettinen *et al.* [20]	✓	–	–	–	–	✓	–	§A.3
Truong *et al.* [36]	✓	✓	✓	–	–	–	–	§A.4
Shrestha *et al.* [28]	–	–	–	✓	✓	–	✓	§A.5

✓ = used; – = not used

2 BACKGROUND

In this section we provide the terminology used in this paper, present our system and threat model, and describe the ZIS schemes that we reproduced and evaluated.

2.1 Terminology

We start by clarifying the relevant terms in the domain of ZIS.

Context information. We define context information as the data collected from device sensors (e.g., microphones, light sensors, etc.), augmented with metadata like timestamps [22].

Context. We refer to a set of context information collected by a device from its ambient environment over time as the context of the device.

Colocation. We define colocation as a set of devices residing in the same physical space. In our scenarios, the spaces are different cars and offices, thus devices within the same car or office are colocated, otherwise non-colocated. The term colocation highly depends on the use case of the ZIS scheme. In the case of wearables, colocated devices are on the same body [4], whereas for a smart home, colocated devices are inside a house [13].

Context feature. We define context feature as a concise context property computed from context information. Context features are based on a snapshot of context information [14, 24, 28, 36] or on relative changes of context information over time [20]. They calculate a distance or similarity metric between two samples of context information [14, 28, 36], or derive a binary fingerprint vector from a sample of context information [20, 24].

2.2 System and Threat Model

We assume an IoT scenario containing a number of devices that are colocated and equipped with a set of sensors to collect context information. The goal of ZIS schemes is to have two colocated devices establish a secure connection (ZIP) or a proof of proximity (ZIA) without user interaction, utilizing context features to secure the process. We assume no established infrastructure and, in the case of ZIP, no prior trust between devices.

Our adversary is based on the models used in the reproduced ZIS schemes. The adversary is an IoT device located in an adjacent car or office. This device can be benign, accidentally trying to pair or authenticate with proximate devices in its wireless range (e.g., IoT device in a neighbor's car), or it can be malicious, intentionally trying to pair or authenticate with non-colocated devices. The adversary is non-colocated with benign devices, thus it can neither observe their context information, nor compromise benign devices to circumvent a ZIS scheme.

However, the adversary is physically close to benign devices (i.e., adjacent car or office), equipped with the same sensing hardware to collect context information, and can communicate with them over a wireless link.

The goal of the adversary is to obtain similar enough context information to fool benign devices into believing that it is co-located with them. Compared to threat models of the reproduced schemes our adversary is more powerful as it possesses two extra capabilities. First, it remains permanently in close proximity to benign devices, including times of low context activity such as during the night. Second, due to symmetric deployment of devices in our scenarios, the adversary has much better chances of following the same trends in context information (e.g., lighting conditions) as benign devices.

We purposely make our adversary powerful to evaluate the scheme performance in challenging scenarios. This allows us to establish the worst-case *baseline adversary*, facilitating comparison of the reproduced ZIS schemes (discussed in Section 5), as well as gain first insights into possible attack vectors for an active adversary [29].

2.3 Reproduced ZIS schemes

To select ZIS schemes for our study, we surveyed frequently cited schemes published at top security venues in the last five years. We selected schemes that targeted IoT scenarios and utilized different context information or context features. We excluded schemes based on behavioral biometrics, e.g., gait [4], gesture [27] or keystroke dynamics [19], as these schemes designed for wearable IoT scenarios. In the end, we arrived at five schemes, which we reproduced from the ground up, relying on the help of the original authors to ensure the correctness of our implementations. We briefly introduce each scheme in its respective result subsection (cf. Section 4) and refer to the Appendix for detailed descriptions.

3 STUDY DESIGN

We designed our study to cover the majority of relevant context information used in current ZIS schemes. We selected three realistic IoT scenarios: in the first two, we used identical sensing devices to collect context information, minimizing the effects of hardware variations on our results. In the third scenario, we used different sensing devices to evaluate the impact of device heterogeneity. This section describes the design and conduct of our experiments, as well as ethical concerns when dealing with sensitive personal data collected in our study.

3.1 Data Collection

The goal of our experiment was to collect a comprehensive real-world dataset of context information that can serve as a baseline for comparing current and future ZIS schemes. In the first two scenarios, we collected data using a *Texas Instruments SensorTag CC2650* and a *Raspberry Pi 3*. Audio data was collected using a *Samson Go* USB microphone, which recorded a mono audio stream with a 16 kHz sampling rate, and encoded it using the lossless *FLAC* format. The Raspberry Pi also collected all visible wireless access points (APs) and BLE devices, including their signal strength, every ten seconds. The remaining context information (accelerometer, barometer, gyroscope, humidity, light intensity, magnetometer, and temperature) was recorded using the SensorTag, connected to the Raspberry Pi using BLE. [1] Sensor data was recorded with a sampling rate of 10 Hz.

In the third scenario, we additionally used *Samsung Galaxy S6* smartphones and *Samsung Gear S3* smartwatches to collect the same context information. Since those devices are not equipped with temperature and humidity sensors we combined them with a *RuuviTag+*. We tried to obtain the same sampling rate on all our devices, however, the Galaxy S6 limits barometric pressure and luminosity readings to 5 Hz. The summary of used sensing devices and sampling rates is given in Table 19 in the Appendix. All events that could influence the context information (e.g., windows/doors being opened or closed, people entering or leaving the recording area, traces of mobile devices, etc.) were documented automatically or by hand in a ground truth sheet.

[1] While accelerometer, gyroscope and magnetometer are not used by any scheme, we collected their data for use in future schemes.

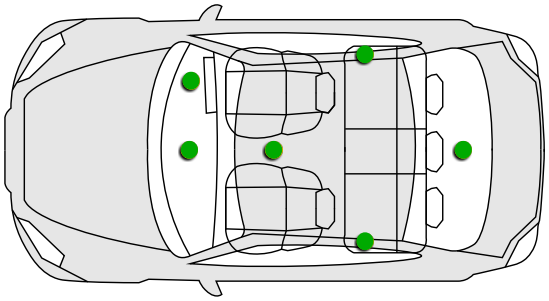


Fig. 1. Device location map in the car scenario

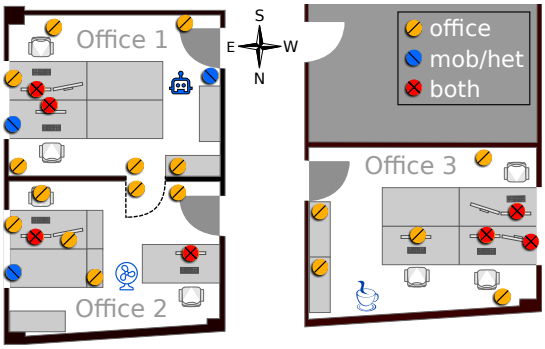


Fig. 2. Device location map in the office and mob/het scenarios

3.2 Scenario 1: Car

In the first scenario, we used two cars from different manufacturers. Each car was equipped with six sensing devices distributed inside the vehicle as shown in Figure 1. The devices occupied similar spots in both cars: one device was placed on top of the dashboard facing the windshield, inside the glove compartment, in between the front seats facing upwards, attached to each handhold above the two rear doors, and put in the middle of the trunk. This placement covers all prominent spots one might expect a sensor or a personal device inside a car (cf. Figure 20 in the Appendix).

After setting up the cars, we drove a predefined route of three hours and 120 km (74 M) on the afternoon of an autumn day. The time was chosen to ensure that the collection began while the sun was visible and ended after sunset, to collect a variety of lighting conditions. The route included city traffic, country roads, and highway (cf. Figure 10 in the Appendix for a map). We drove both cars close to each other within a distance of 20 m (65 ft), which we varied from time to time. In addition, we took a short break, with the cars parked side by side.

The challenge for the ZIS schemes is to identify colocated devices in a single car, while excluding devices from different cars that might be nearby or just listening to the same radio station.

3.3 Scenario 2: Office

A typical application for IoT devices is the deployment in a smart home or office. To collect realistic context information in this scenario, we deployed eight sensing devices in three office rooms as shown in Figure 2. We put the devices in similar places, representing typical IoT spots: one device was attached to the main screen of a workplace (smart workstation, several spots), above the windows (smart shades), near the ceiling lights (smart lights), in a closed cupboard (smart device), near the door at around two meters height (smart room/motion sensor) and in a corner at around 2.5 meters (environmental sensor). The summary of device locations in the office scenario is given in Table 21 in the Appendix.

We collected context information for one full week, resulting in five work days with people present and two days of the weekend, when offices 1 and 2 were empty, and one person was working in office 3. Offices 1 and 2 were adjacent and connected with a door, which was closed most of the time. Office 3 was on the opposite side of the floor. All three rooms had a similar setup in terms of size and position of furniture but a different number of participants working in them (one in office 2, two in office 1 and three in office 3).

The collected dataset is intended for testing ZIS schemes designed for smart homes and offices. The challenge here is to distinguish between the three different rooms. Ideally, a scheme identifies all colocated devices in one room but excludes all others.

3.4 Scenario 3: Office with Mobile Heterogeneous Devices (Mob/het)

We extended the office scenario by including both static devices permanently residing inside offices, and mobile devices carried by users (cf. Figure 2). We added a number of appliances (i.e., vacuum robot and its station in office 1, fan in office 2, coffee machine in office 3), facilitating device mobility when users move to use them. Each office was equipped with four static devices (SensorTag), covering similar spots and the appliances: one device was attached to the main screen of a workplace (smart workstation, several spots), near a power plug (smart plug), on top of a vacuum robot station (smart robot station) and coffee machine (smart coffee maker), near a fan (smart fan). We equipped four participants with three mobile devices each: a laptop (with attached smartphone to collect context information), smartphone and smartwatch. We also placed a smartphone on top of the robot vacuum cleaner. Device locations are summarized in Table 22 in the Appendix.

We collected context information for eight hours from 9 am till 5 pm, representing a typical working day. Over the course of the day participants moved freely between the offices to get a cup of coffee, have a meal or attend a meeting, each time carrying a set of their mobile devices. We also moved the vacuum robot between the offices, letting it autonomously run a complete cleaning cycle.

Similarly to the office scenario, the challenge for ZIS schemes is to distinguish devices present in the same office, while excluding devices in others.

3.5 Reproducibility and Re-usability

In total, our dataset contains 239 GB of context information, including more than 4250 hours of audio recordings, over 1 billion sensor readings, and over 12 million WiFi and BLE beacons. Computing the context features of the reproduced schemes took over 300 000 CPU hours. The audio-based features were computed using Matlab on a high-performance cluster. The remaining features were implemented in Python on a high-performance server. After compression, they utilize almost 1 TB of disk space. This includes the computed features, aggregated statistics, and metadata for reproduction and validation following the recommendations by Benureau *et al.* [2].

To facilitate future reuse, we release the source code of the entire data collection and evaluation stack, as well as the collected context information in an anonymized form, all intermediate and final data files (including machine learning models) and the code used to generate the visualizations. Privacy concerns prevent us from releasing the audio data recorded in the Car and Office scenarios, but we are able to provide researchers with the audio recordings from the Mobile scenario upon request [8]. See [9] for an index of all released data and code.

3.6 Ethical Considerations

The study was approved by our institutional ethical review board, data protection officer, and workers' council. Participants gave informed consent for the collection, use, and release of the data. During collection, the audio data was encrypted with keys controlled by the affected participants, requiring their explicit consent and cooperation to decrypt the data for processing. In the mob/het scenario, we gave participants the chance to inspect the recordings before obtaining informed consent for their release.

4 RESULTS

In this section, we report on the performance in distinguishing colocated and non-colocated devices for the five reproduced schemes (cf. Table 2). The performance evaluation of each scheme is structured as follows. We first provide a concise overview of the scheme by explaining the context features used to distinguish colocated and non-colocated devices. Then, we explain the methodology of the original scheme and provide details of our evaluation. Next, we present and interpret the performance results of the scheme for each scenario. To quantify the performance we compute the Equal Error Rate (EER), which is the point of equal False Accept Rate (FAR) and

Table 2. Overview of the reproduced schemes and evaluation result

Scheme	Sect.	Conclusion	Best EER		
			Car	Office	Mob/het
Karapanos *et al.* [14]	4.1	Best EER for car scenario among the reproduced schemes. Limited robustness on intervals from 5 to 15 seconds. Breaks down in heterogeneous setting.	0.006	0.098	0.157
Schürmann, Sigg [24]	4.2	Generates fingerprints with good randomness properties, but shows varying performance on subscenarios, and provides only limited robustness.	0.154	0.241	0.140
Miettinen *et al.* [20]	4.3	Insufficient fingerprint randomness leads to some error rates exceeding 0.5. Low robustness. Audio-based fingerprints perform better than luminosity-based fingerprints.	0.226	0.120	—
Truong *et al.* [36]	4.4	Achieves the best error rates in office and mob/het scenario, but shows low robustness and high reliance on audio feature, and struggles with heterogeneous settings.	0.104	0.069	0.123
Shrestha *et al.* [28]	4.5	Promising performance in the car scenario, but lower performance in the office and mob/het scenario, low robustness, and high redundancy and ambiguity in features.	0.115	0.247	0.141

False Reject Rate (FRR). In addition, we assess how much usability the schemes can deliver if a specific security level is required by setting a number of target FARs (between 0.1% and 5%) and analyzing the resulting FRRs.

We evaluate the scheme robustness by analyzing an increase in error rates (either FAR or FRR) from the original EER when applying parameters found to be optimal in one scenario to another. This simulates a scheme being used in a scenario it was not trained on, like an IoT device optimized for office use being deployed in a car. We further summarize each studied scheme by comparing our results with the original findings and providing key takeaways from our evaluation. This facilitates a direct comparison of the different schemes in our scenarios.

We introduce subscenarios to investigate the impact of changes in the environment (e.g., time of day, moving vs. parked cars) on the scheme performance. A subscenario represents a subset of context information collected at a specific stage in the scenario. For the car scenario, we distinguish three subscenarios: the *city* and *highway* subscenarios contain context information of the cars driving inside city limits or on the highway, respectively, and the *parked* subscenario includes context information from the time the cars were parked. Similarly, we construct three subscenarios for the office scenario: the *weekday* subscenario contains context information collected from Monday to Friday from 8 am to 9 pm, the *night* includes context information for all seven days from 9 pm to 8 am, and the *weekend* consists of context information from Saturday and Sunday in the timeframe from 8 am to 9 pm. We omit the subscenario evaluation in the mob/het scenario as there were no specific stages in this scenario.

We assess the performance of all schemes except [36] and [28] on time intervals of 5, 10, 15, 30, 60 and 120 seconds with the length denoted t. The interval represents a timeframe over which context information is aggregated to compute a context feature, e.g., a 5 second audio snippet or a 30 second WiFi capture. [36] is evaluated on time intervals of 10 and 30 seconds, as the scheme is less well-suited to an arbitrary interval length due to the used features, while [28] does not use any intervals.

Table 3. EER summary for Karapanos *et al.*

	Car				Office				Mob/het
t	Full	City	Highway	Parked	Full	Night	Weekday	Weekend	Full
5	0.050	0.071	0.009	0.124	0.141	0.140	0.135	0.143	0.157
10	0.032	0.049	0.003	0.071	0.133	0.132	0.128	0.136	0.168
15	0.026	0.043	0.002	0.060	0.128	0.126	0.123	0.129	0.170*[2]
30	0.017	0.031	0.001	0.022	0.118	0.115	0.116	0.115	0.172
60	0.008	0.014	0.002	0.007	0.107	0.102	0.109	0.099	0.179*
120	0.006	0.010	0.000	0.037	0.098	0.090	0.103	0.081	0.183*

4.1 Karapanos *et al.*

Karapanos *et al.* [14] proposed using maximum cross-correlation between snippets of ambient audio from two devices to decide if they are colocated. The cross-correlation is computed on a set of one-third octave bands [1] and averaged to a similarity score. One-third octave bands split the audible spectrum (20 Hz to 20 kHz) into 32 frequency ranges of different sizes. To prevent erroneous authentication when audio activity is low, a power threshold is applied to discard audio snippets with insufficient average power. The similarity score is checked against a fixed similarity threshold to decide if two devices are colocated, and can thus be authenticated. Tuning the similarity threshold allows trading usability for security and vice versa. The authors evaluated their scheme in several scenarios such as a quiet office, lecture hall, and café. The scheme details are given in Appendix A.1.

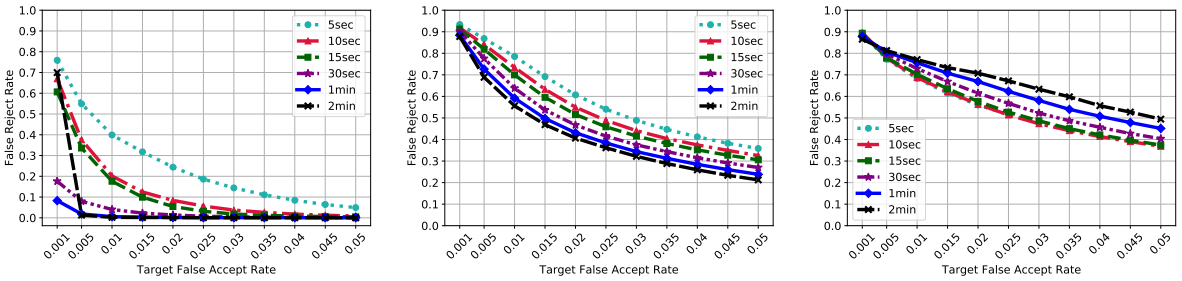
4.1.1 Methodology. To investigate the scheme performance we compute similarity scores between colocated and non-colocated devices on different interval lengths. We increase the minimum length of audio snippet and maximum correlation lag to achieve a comparable level of synchronization to the original implementation. These changes have a negligible impact on the similarity score computation, as stated in Appendix A.1. To understand factors affecting the performance, we analyze the behavior of similarity scores on different octave bands.

4.1.2 Car. We observe EERs between 0.006 and 0.050, decreasing with rising interval length (cf. Table 3). To understand this behavior we compute the distributions of colocated and non-colocated similarity scores for each interval. Overlaps of these distributions explain the corresponding error rates: in the car scenario, the overlaps range from 1.1% to 8.5%. We observe a clearer separation between colocated and non-colocated similarity scores at longer intervals, caused by a sharper drop of non-colocated similarity scores. When targeting low FARs, the resulting FRRs are below 0.2 on the intervals above $t = 15$, dropping rapidly with a growing FAR (cf. Figure 3a).

Our octave band analysis shows the profound influence of lower frequencies (below 315 Hz) caused by a running car on the overall similarity score. This explains the lowest EERs reaching 0.0 in the uniform sound environment of a highway (cf. Table 3). The more diverse sound environment of a city shows a severalfold increase in EERs compared to the highway subscenario. Surprisingly, in a low-activity environment of parked cars, the EERs are only a few percentage points above the city subscenario. Investigating this phenomenon revealed that the power threshold discards up to 90% of similarity scores in the parked subscenario, retaining only those scores that resulted from intense audio activity.

Applying office and mob/het EER thresholds to the car dataset leads to a marginal increase in error rates below 1 percentage point on the intervals $t = 5$ to 15 for the office, and on $t = 10, t = 15$ for the mob/het, with other intervals showing severalfold growths in error rates. Among subscenarios, we see limited robustness between

[2]In cases where FAR and FRR do not match to three digits after the decimal, we average them and denote the result as EER*.



(a) Car (b) Office (c) Mob/het

Fig. 3. FRRs with target FARs for Karapanos *et al.* in the full car, office and mob/het scenarios

quiet (parked) and active environments (city and highway) at $t = 120$, as well as when applying city thresholds to the highway dataset for $t = 60, t = 120$.

4.1.3 Office. In the office scenario, we observe EERs between 0.098 and 0.141, decreasing with growing interval length (cf. Table 3). We attribute these EERs to larger overlaps between colocated and non-colocated classes, ranging from 19% to 28%. We see a clear trend of higher similarity scores between non-colocated devices in adjacent offices (offices 1 and 2 in Figure 2). Our octave band analysis reveals close resemblance between these scores on lower frequencies below 250 Hz and on higher frequencies above 1250 Hz. Thus, both low frequencies penetrating adjacent offices and high frequency sounds like a police siren can increase non-colocated similarity scores. When targeting low FARs, the resulting FRRs start around 0.9 and never drop below 0.2 (cf. Figure 3b).

We observe that higher audio activity of weekdays results in lower EERs on the intervals below $t = 30$. However, on longer intervals the EERs of low-activity environments (i.e., night and weekend) become lower compared to the weekday. Investigating this phenomenon in more detail reveals two reasons for such a behavior. First, the power threshold retains a few similarity scores originated from intense audio activity in the night and weekend subscenarios. Second, in low-activity environments sounds are infrequent, localized and short-term, making them easier to capture on longer intervals by colocated devices and less prone to be leaked to non-colocated devices.

Applying car and mob/het EER thresholds to the office dataset results in a minor increase in error rates below 2 percentage points on the intervals $t = 10, t = 15$ for the car, and on $t = 10$ for the mob/het, with other error rates rising a few extra percentage points. In subscenarios, we observe robustness only between low-activity environments of night and weekend, showing an increase in error rates below 2 percentage points on all intervals.

4.1.4 Mob/het. Investigating similarity scores in the mob/het scenario revealed that 75% to 100% of the scores generated by smartphones and watches were discarded by the default power threshold (40 dB) in the absence of intense audio activity (e.g., running vacuum robot). We adjusted the power thresholds for smartphones and watches to 38 and 35 dB respectively, significantly increasing the scheme availability in the cases of medium audio activity (e.g., low-voiced conversation), while still discarding the similarity scores from quiet environments.

With the new power thresholds, we observe increased EERs between 0.157 and 0.183, rising with interval length, reversing the trend seen in the car and office scenarios (cf. Table 3). Once again, higher EERs are explained by larger overlaps between colocated and non-colocated classes, ranging from 33% to 36%. When targeting low FARs, the resulting FRRs vary almost linearly between 0.9 and 0.37 (cf. Figure 3c).

We found that microphone diversity and device mobility are likely reasons for the reversed EER trend. The similarity scores among heterogeneous devices are generally lower, decreasing significantly towards longer intervals. Our analysis suggests that the main reason for these lowered scores is diverse sensitivity and frequency response

of heterogeneous microphones [15]. We empirically observed that smartwatch microphones are optimized for human voice but rather insensitive to low frequencies, while on smartphones low frequencies cause a lot of noise in recordings, and the USB microphones show the best signal quality on a wide frequency range. On longer intervals, device mobility further increases signal variation: the probability of capturing a unique signal (e.g., a keystroke by smartwatch) or wide-band scratching noises (e.g., smartphone rubbing against a pocket) increases.

Applying car and office EER thresholds to the mob/het dataset leads to a minor increase in error rates up to 1.5 percentage points on the intervals $t = 10, t = 15$ for the car, and $t = 10$ for the office, with other intervals showing several percentage points extra growths in error rates.

4.1.5 Conclusion. Our results show that the scheme by Karapanos *et al.* can reliably distinguish colocated and non-colocated devices in the car scenario, but degrades in performance in the office and mob/het. We generally achieve higher EERs compared to the authors, who observe an EER of 0.002. Possible reasons for that are the increased distance between colocated devices and sustained closeness of non-colocated devices in our scenarios.

When the scheme is used among homogeneous devices (car and office scenarios) we observe better performance with increasing interval length and more intense audio activity. The difference between car and office EERs is due to a smaller distance between colocated devices in the car, and more intense audio activity, especially on lower frequencies (highway). We see that highway EERs decrease marginally towards longer intervals, suggesting the use of short- to medium-sized intervals in active environments, reducing the run-time overhead of the scheme.

With heterogeneous devices (mob/het scenario) using longer intervals decreases the scheme performance, and intense audio activity is only beneficial if heterogeneous microphones can similarly record it (e.g., human voice), otherwise the performance will further decrease, especially on longer intervals. Considering that built-in microphones in mobile devices are user-interaction oriented, the scheme can benefit from shorter intervals and audio activity in the frequency range of human voice in heterogeneous settings.

The power threshold allows the scheme to cope with quiet environments, sometimes at the price of excluding a significant portion of the dataset (e.g., parked car), trading off availability for security. However, as we have seen in the mob/het scenario, the power threshold proposed by the authors severely decreases scheme availability already in the cases of medium audio activity, urging the need to carefully select this parameter, depending on the characteristics of the microphones.

The scheme consistently shows robustness on medium-sized intervals ($t = 10, t = 15$) among our scenarios, suggesting that it can potentially adapt to new environments on these intervals.

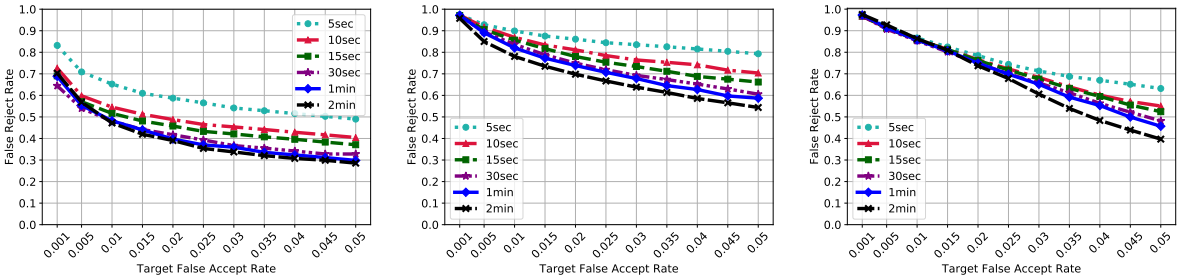
4.2 Schürmann and Sigg

Schürmann and Sigg [24] propose encoding a snippet of ambient audio into a binary fingerprint to pair two devices. The generated fingerprint consists of 16 individual shorter fingerprints that reflect the energy changes of successive frequency bands in the audio snippet over shorter timeframes. The similarity between the fingerprints derived by two devices informs a pairing decision. These fingerprints need to exhibit good randomness in order to secure a key establishment procedure between devices via fuzzy commitments. The authors evaluated their scheme in a series of deployments, ranging from staged lab measurements to recordings in a busy canteen and near a road. A detailed description of the scheme can be found in Appendix A.2.

4.2.1 Methodology. We evaluate the performance of the scheme by generating fingerprints using different intervals t. Due to hardware constraints, we use a lower audio sampling rate, which reduces the length of the fingerprint from 512 to 496 bits. This change introduces a marginal deviation from the original implementation as detailed in Appendix A.2. To evaluate the similarity of the generated fingerprints of two devices, we calculate the similarity percentage as $1 - (hamming\_dist/length)$.

Table 4. EER* summary for Schürmann and Sigg

	Car				Office				Mob/het
t	Full	City	Highway	Parked	Full	Night	Weekday	Weekend	Full
5	0.271	0.228	0.247	0.362	0.419	0.423	0.406	0.440	0.363
10	0.226	0.175	0.199	0.359	0.351	0.365	0.319	0.380	0.257
15	0.211	0.157	0.170	0.361	0.317	0.340	0.267	0.347	0.215
30	0.179	0.121	0.126	0.361	0.277	0.308	0.215	0.309	0.175
60	0.160	0.100	0.106	0.359	0.256	0.287	0.194	0.280	0.154
120	0.154	0.096	0.112	0.328	0.241	0.275	0.178	0.253	0.140



(a) Car (b) Office (c) Mob/het

Fig. 4. FRRs with target FARs for Schürmann and Sigg in the full car, office and mob/het scenarios

The scheme uses a fixed similarity threshold that distinguishes colocated from non-colocated devices. In addition, we investigate the randomness of the fingerprints by interpreting them as random walks, with 1- and 0-bits representing steps in the positive and negative direction [4]. The outcomes will follow a binomial distribution if the fingerprints are uniformly random. We also investigate bit transition probabilities by interpreting each bit of the fingerprint as a state in a Markov chain.

4.2.2 Car. We observe EERs between 0.154 and 0.271, decreasing with increasing interval length t (cf. Table 4). These error rates correspond to the observed overlaps in similarity between colocated and non-colocated devices, which ranges between 30% and 51%. When optimizing for a low FAR, the resulting FRRs exceed 0.8 for certain parameters and never drop significantly below 0.3 (cf. Figure 4a). The system performs best in scenarios with diverse sound environments, like driving within city limits, showing consistently lower EERs in the city subscenario (cf. Table 4), dropping as low as 0.096. Environments with a uniform sound environment, like driving on the highway, show slightly increased error rates, but still remain consistently below the error rates for the full dataset. In low-activity environments like parked cars, the scheme shows significantly increased error rates of up to 0.362—an increase of 0.134 over the city environment with the same parameters.

The fingerprints exhibit good randomness across all devices. Their Markov property is good, with $P(b = 1) \approx 0.5$ for all bits. When interpreting fingerprints as random walks, the resulting distribution of endpoints is close to the expected binomial distribution (cf. Figure 5). When splitting the 496-bit fingerprints into their constituent 31-bit fingerprints and analyzing them separately, the random walks show a more varied distribution. Some are close to the expected binomial distribution (cf. Figure 5d), while others show a flatter distribution (cf. Figure 5c),

indicating more fingerprints contain a larger number of 1- or 0-bits than expected. Investigating these sensors in more detail, we found that their microphones were affixed to surfaces that vibrated more than average. As fingerprints are derived from variations in signal energy over time, the biased fingerprints may have been caused by periodic variations in the energy induced by the vibrations.

Applying the threshold from the office scenario to this dataset results in an increase in error rates between 3.6 and 11.2 percentage points, with the larger changes occurring for $t = 5$ and 120. The mob/het threshold increases the error rates by 1.7 to 7.1 percentage points, with the largest changes for $t = 15$ and 30, while $t = 120$ shows the smallest change. In subscenarios, the most stable results are obtained between city and highway, changing between 4.1 and 9.5 percentage points in both directions. The other combinations show significantly larger error rate increases, in some cases up to 25.7 percentage points. This indicates that the scheme has limited robustness in cases where the environments are similar, but is not robust to larger changes in environmental characteristics.

4.2.3 Office. In the office, we observe generally increased EERs, ranging from 0.241 to 0.419 and decreasing with increasing interval lengths (cf. Table 4). These error rates are explained by the higher overlaps between colocated and non-colocated classes, which lie between 48% and 79%. In particular, we observe that the computed similarities between some non-colocated devices exceeded the similarities with all of their respective colocated devices, especially using smaller interval sizes t. Investigating these anomalous pairs in more detail revealed that the high similarities occur mostly at night and on the weekend, i.e., at times of very low ambient activity. However, the question why these particular devices were affected while others behaved normally remains unanswered.

When optimizing for a low FAR, the resulting FRRs for the full scenario are universally above 0.5 (cf. Figure 4b). Once again, the system performs best in environments with high audio activity, in this case the weekdays, showing significantly reduced error rates compared to the night and weekend.

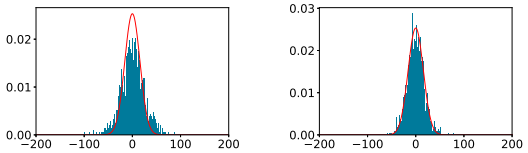
The fingerprints again show good randomness, with a strong Markov property and random walks close to the expected distribution for the full fingerprints. When investigating the sub-fingerprints, we observe a slight bias towards 0 in the lowest three bits of some devices, with $P(b = 1) \approx 0.48$. Most of the affected devices were located in office 2, but there is no discernible pattern in which devices exhibit this behavior and no obvious explanation.

Applying the car threshold to this dataset results in error rate increases of 3.9 to 10.9 percentage points, with the largest changes at $t = 5$ and 120. Conversely, the threshold obtained in the mob/het scenario will increase error rates by 5.6 to 12.6 percentage points, with the largest changes at $t = 10$, 15 and 30. The error rates of the night and weekend subscenarios remain almost completely stable when exchanging their thresholds. All other combinations show larger changes, often showing swings of more than 10 percentage points.

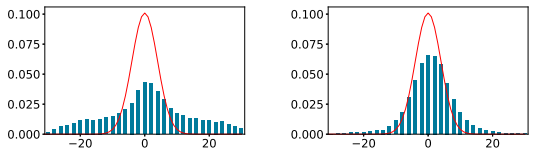
4.2.4 Mob/het. The error rates in the mob/het scenario exhibit a larger spread than in the other scenarios, with EERs ranging from 0.140 to 0.363 and decreasing with rising interval lengths (cf. Table 4). They once again correlate with the overlaps in similarity between colocated and non-colocated devices, which ranges from 27% to 62%. When optimizing for low FARs, the resulting FRRs range from close to 1.0 to 0.40 (cf. Figure 4c).

Although the Markov property is universally good, the randomness of the fingerprints shows significant variation. While the fixed sensors show decent randomness, the mobile devices (smartphones and smart watches) deviate from the expected distribution, showing similar behavior to the biased sensors in the car scenario. Part of this deviation can likely be explained by the different characteristics of the microphones (cf. Section 4.1.4). Devices that were covered (i.e., smartphones in pockets and smart watches worn under long-sleeved clothing) showed the largest deviation from the expected distribution, with strong biases towards sub-fingerprints consisting of mostly 1- or 0-bits. This is likely related to the movement of cloth over the devices generating wide-band scratching noises, in combination with sound attenuation caused by the clothing.

Applying the car threshold to this dataset results in increases in error rates between 1.5 and 7.4 percentage points, with the largest changes for $t = 15$ and 30. The office threshold increases error rates by 5.3 to 11.8 percentage points, with the highest increases for $t = 15$ and 30.

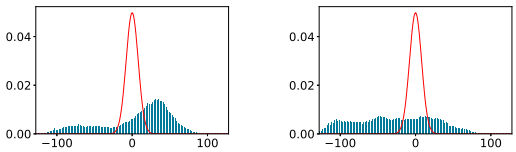


(a) Full fingerprint, Device A (b) Full fingerprint, Device B

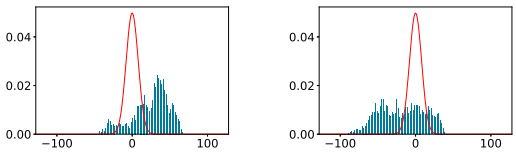


(c) Sub-fingerprint, Device A (d) Sub-fingerprint, Device B

Fig. 5. Distribution of fingerprint random walks for Schürmann and Sigg, Car scenario, $t = 10$. Expected binomial distribution in red.



(a) Device A, $t = 5$ (b) Device B, $t = 5$



(c) Device A, $t = 120$ (d) Device B, $t = 120$

Fig. 6. Distribution of fingerprint random walks of two representative devices for audio feature of Miettinen *et al.*, Office, $b = 128$. Expected binomial distribution in red.

4.2.5 Conclusion. The scheme by Schürmann and Sigg is unable to reliably distinguish colocated from non-colocated devices in our scenarios. We also observed unexplained high similarities for specific non-colocated device pairs in the office scenario. In particular, the scheme breaks down in environments with low ambient activity (a limitation also noted by the original authors), however, even in high-activity environments like a driving car, the error rates exceed 10% for almost all parameters. Still, it may be possible to increase the overall performance of the scheme by excluding low-energy samples with a power threshold (similar to Karapanos *et al.*).

The fingerprints exhibit good randomness in many cases, however, they struggle with noisy inputs, like vibration- or friction-induced sounds, and will in some cases generate fingerprints that consist almost entirely of 1- or 0-bits. In particular, devices carried in pockets or under long sleeves seem to cause problems.

While the scheme is robust in some pairs of subscenarios, the robustness is very limited. Interestingly, the intervals behave differently for different combinations of scenarios—while the error rates of $t = 120$ are almost unaffected in some pairs, in others, they show very large changes. The same is true for other intervals like $t = 15$.

Schürmann and Sigg do not report error rates in their evaluation, so a direct comparison is impossible. However, the average separation between colocated and non-colocated fingerprints they report is larger than that observed in our scenarios. One possible explanation may be a tighter synchronization of audio signals in their experiment, as their samples were recorded by a single device with two microphones, thus avoiding any problems related to recordings not being exactly in sync. In a practical setting, such a tight synchronization between two devices will be more challenging to achieve (our synchronization method is described in Appendix A).

4.3 Miettinen *et al.*

The scheme proposed by Miettinen *et al.* [20] uses two context features, one based on audio and the other on luminosity. In both cases, changes over extended timeframes are recorded and encoded into a binary *context fingerprint* of a fixed length b. The similarity of these fingerprints is then used to decide if devices can establish a connection by serving as a shared secret to bootstrap a key exchange using fuzzy commitments. Due to this usage, the randomness of the fingerprints is once again of interest. The authors evaluated their scheme in an office, a home scenario, and a mobile scenario simulating wearable devices. They also propose an optional extension to

Table 5. EER* results for Miettinen *et al.*

		Audio						Luminosity					
		$t = 5$	10	15	30	60	120	5	10	15	30	60	120
Car	$b = 64$	0.377	0.389	0.396	0.384	0.382	0.263	0.506	0.505	0.504	0.505	0.501	0.517
	128	0.358	0.368	0.370	0.372	0.362		0.507	0.506	**0.492**	0.499	0.516	
	256	0.329	0.335	0.328	0.295			0.505	0.499	0.498	0.514		
	512	0.344	0.294	0.287				0.497	0.504	0.517			
	1024	0.297	**0.226**					0.498	0.522				
Office	64	0.249	0.228	0.218	0.204	0.202	0.193	0.495	0.491	0.486	0.468	0.444	0.425
	128	0.226	0.206	0.203	0.190	0.184	0.172	0.487	0.477	0.469	0.447	0.418	0.406
	256	0.212	0.196	0.193	0.180	0.165	0.147	0.483	0.470	0.459	0.421	0.397	0.403
	512	0.203	0.188	0.185	0.166	0.136	0.131	0.471	0.454	0.440	0.397	0.362	0.400
	1024	0.197	0.184	0.178	0.135	**0.120**	0.126	0.454	0.437	0.426	**0.344**	0.363	0.362
Mob/het†	64	0.377	0.368	0.364	0.383	0.349	0.314	**0.517**	0.520	0.520	0.521	0.525	0.524
	128	0.356	0.344	0.339	0.371	0.325		0.521	0.523	0.522	0.525	0.519	
	256	0.331	0.322	0.305	0.365			0.521	0.528	0.520	0.524		
	512	0.306	0.308	0.291				0.522	0.526	0.518			
	1024	**0.287**						0.525					

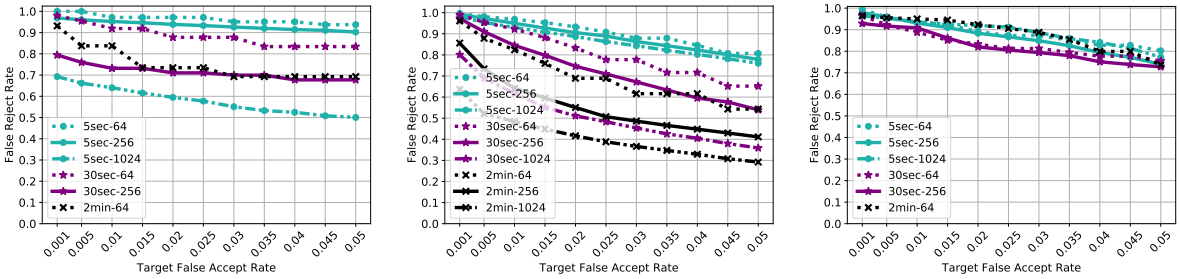
Empty cell denotes insufficient data to generate fingerprint. Best value in scenario marked in **bold**.
† Computed on subset.

ensure sufficient fingerprint quality by discarding fingerprints with an insufficient *surprisal*, which measures how unexpected a fingerprint is for the current time of day. However, they did not evaluate the effect of this proposal. More details are given in Appendix A.3.

4.3.1 Methodology. Our methodology is identical to that used for the paper by Schürmann and Sigg (cf. Section 4.2). As the fingerprints generated by the scheme span long timeframes (up to 34 hours), we omit the subscenario evaluation, as allocating fingerprints to specific subscenario timeframes is impossible.

4.3.2 Car. Both luminosity- and audio-based fingerprints show relatively high error rates, with the lowest observed EER* being 0.492 and 0.226, respectively (cf. Table 5). These high error rates can be explained by the high overlap of similarity percentages between the colocated and non-colocated groups, showing overlaps between 83% and 96% for the luminosity fingerprints. The overlaps are lower, but still significant for the audio fingerprints, with overlaps between 39% and 79% being observed. When aiming for a specific FAR, the resulting FRR is universally above 0.5 for the audio fingerprint (cf. Figure 7a). For the luminosity feature, the FRRs are 1.0 for all targeted FARs, indicating that all samples are rejected, making the scheme usability unacceptable.

Once again, the security of the scheme does not only depend on the error rates, but also on the randomness of the generated fingerprints. Here, we observe the luminosity fingerprints to be heavily biased towards zero. The audio fingerprints contain more 1-bits, but still do not show sufficient randomness. This limited randomness and high bias also explain the high overlap in the fingerprint similarity distributions. Rejecting fingerprints with insufficient *surprisal* excluded over 90% of the luminosity fingerprints even for the smallest specified surprisal value and consistently increased error rates for all attempted thresholds. For audio fingerprints, we evaluated



(a) Audio feature, Car (b) Audio feature, Office (c) Audio feature, Mob/het

Fig. 7. FRRs with target FARs for Miettinen *et al.* in the full car, office and mob/het scenarios for a selection of parameters.

a series of thresholds for different parameters and found that in many cases, the error rates do not decrease significantly and in some cases will even increase, unless over 95% of the dataset is excluded.

Applying the threshold from the office scenario increases the error rates for audio fingerprints by varying amounts, in some cases remaining stable, in others increasing by close to 25 percentage points, where higher values of b and t result in higher robustness. For luminosity fingerprints, increasing b reduces robustness and can lead to all samples being rejected, while smaller values of b with large t sometimes show stable error rates. With the mob/het threshold, the system rejects all audio fingerprints. On luminosity fingerprints, it shows unpredictable behavior, being robust for certain parameters and rejecting all samples for others, with no discernible patterns.

4.3.3 Office. In the office scenario, we observe lower error rates, with EERs between 0.249 and 0.120 for the audio fingerprints (cf. Table 5), which can be explained by the decreased overlaps between the fingerprint similarity percentages of colocated and non-colocated devices (between 24% and 49%). For the luminosity fingerprints, the error rates remain high, with the lowest observed EER being 0.344, which can be explained by overlaps between 80% and 99%. In many cases, FAR and FRR only become equal with thresholds close to 100% similarity, at which point the FAR becomes 0.0 and the FRR 1.0. When aiming for a low FAR, the resulting FRRs remain large for both audio (cf. Figure 7b) and luminosity fingerprints (where the error rate is almost universally 1.0).

The luminosity fingerprints consist overwhelmingly of 0-bits, which explains the observed overlaps in similarity percentages. This can be explained by the low variance in luminosity in offices, which are often lit by electric lighting with only very infrequent changes. Audio fingerprints show more variance but are usually biased, with the probability of obtaining a 1-bit varying between 0.4 and 0.63. The distribution within the fingerprint is also unequal, as the fingerprints are almost completely zero at night, leading to further biases (cf. Figure 6). Rejecting fingerprints with insufficient surprisal once again excluded most of the dataset in the luminosity feature, and only led to improvements of 1-2 percentage points in the audio fingerprints while excluding 10-20% of the dataset.

Applying the car threshold to the dataset again results in varying increases in the audio fingerprint error rates, following the same trends outlined for this combination in the car section. The luminosity feature accepts almost all fingerprints, with FARs between 0.9 and 1.0. With the threshold from the mob/het scenario, the scheme rejects all audio fingerprints, and either accepts or rejects all luminosity fingerprints, following no discernible pattern.

4.3.4 Mob/het. In the mob/het scenario, the colocation of devices changes over time, as they move between offices. This makes it impossible to perform a comprehensive evaluation of the scheme proposed by Miettinen *et al.*, as the mobile devices often do not stay colocated with any device long enough to establish a pairing. We thus limit our evaluation to a timeframe of approx. 2.5 hours at the beginning of the recording, during which the colocation of all devices remains static.

The error rates for both luminosity and audio fingerprints are increased compared to the other scenarios, in some cases significantly so (cf. Table 5). The EERs of the luminosity fingerprints are above 0.5 for all combinations of parameters, and the best observed EER of the audio fingerprint exceeds those of the car and office scenarios by more than four percentage points. Aiming for a low FAR will result in unacceptably high FRRs (cf. Figure 7c). For audio fingerprints, this decreased performance can be attributed to the varying microphone characteristics leading to sounds being received with different amplitudes, resulting in deviating fingerprints. The luminosity fingerprints are challenged by the different positions of the mobile devices, which are in some cases carried in pockets and thus do not receive the same luminosity readings as other devices.

Luminosity fingerprints remain heavily biased towards zero, and the audio fingerprints also frequently show strong biases towards 1 or 0, following no discernible dependence on the parameters t and b. Using the surprisal thresholds leads to small improvements (less than 2 percentage points) in the error rates for audio fingerprints, at the cost of excluding 10-20% of the dataset. For luminosity fingerprints, even the smallest threshold excludes 96% of the dataset and does not improve the error rates significantly.

Applying the car threshold to the dataset will result in varying error rates, often rejecting all samples, and never coming close to the original error rates for the audio fingerprint. The luminosity fingerprints will occasionally reach error rates close to the original, following no particular pattern, but will often reject all fingerprints as well. The behavior of the office threshold is similar, rejecting close to all samples for fingerprint types.

4.3.5 Conclusion. Our evaluation has shown that the scheme is unable to provide good separation between colocated and non-colocated devices, exhibiting large FARs and FRRs. Low FARs can only be obtained at the cost of large FRRs. The best performance is achieved using audio fingerprints in the office scenario, likely because of the homogeneous hardware and low level of background noise. We also investigate the impact of using the surprisal thresholds proposed by Miettinen *et al.* and find that it will in some cases slightly increase the performance of the scheme but excludes a significant fraction of the dataset in the process, reducing the availability.

The randomness of the generated fingerprints is limited, with devices often showing strong biases towards either 1 or 0, enabling adversaries to break the scheme in a practical deployment by guessing the fingerprint. This illustrates the importance of using an environmental data source with sufficient variability (unlike fixed electric lighting) and a quantization scheme that ensures a roughly equal proportion of 1- and 0-bits, e.g., [24].

Miettinen *et al.* did not compute error rates but observed an average colocated luminosity and audio fingerprint similarity of 95% and 91.8%, respectively, using an interval of $t = 120$ in their office scenario. For non-colocated devices, they saw similarities between 68% and 88% for luminosity and 62% to 71% for audio. We were unable to achieve this degree of similarity on our dataset.

4.4 Truong *et al.*

Truong *et al.* [36] propose combining multiple types of context information to increase the reliability and performance of ZIA schemes. They collect WiFi, Bluetooth, GPS, and audio data and compute a number of context features, aggregated over a time interval t. Features based on the first three modalities are computed based on distances between sets of observed devices and signal strengths, while the audio data is used to calculate the maximum cross-correlation and time-frequency distance between the audio snippets. Colocation is determined using a machine learning classifier, which has been trained with a labeled dataset of colocated and non-colocated features. Due to technical limitations of the used hardware, we were unable to capture GPS data. However, Truong *et al.* found that the GPS feature contained the least amount of discriminative power in their dataset, which was obtained by having volunteers in two cities collect context information and colocation ground-truth data using smartphones and tablets in locations of their choice. The full details of the scheme are given in Section A.4.

Table 6. Classification results for Truong *et al.*, Car

Scenario	t	Model	EER	AUC	Acc.
Car	10	GBM	0.111	0.961	88.8%
– City	10	GBM	0.038	0.993	96.1%
– Highway	10	GBM	0.026	0.995	97.4%
– Parked	10	GBM	0.271*	0.813	72.9%
Car	30	GBM	0.104*	0.967	89.6%
– City	30	GBM	0.032	0.995	96.8%
– Highway	30	GBM	0.022	0.997	97.7%
– Parked	30	GBM	0.282*	0.803	71.7%

Table 7. Classification results for Truong *et al.*, Office

Scenario	t	Model	EER	AUC	Acc.
Office	10	GBM	0.084*	0.974	91.5%
– Night	10	GBM	0.08*	0.976	91.9%
– Weekday	10	DRF	0.087	0.973	91.3%
– Weekend	10	GBM	0.071	0.981	92.9%
Office	30	GBM	0.069	0.982	93.1%
– Night	30	GBM	0.063*	0.984	93.6%
– Weekday	30	GBM	0.072*	0.981	92.8%
– Weekend	30	GBM	0.053	0.989	94.6%

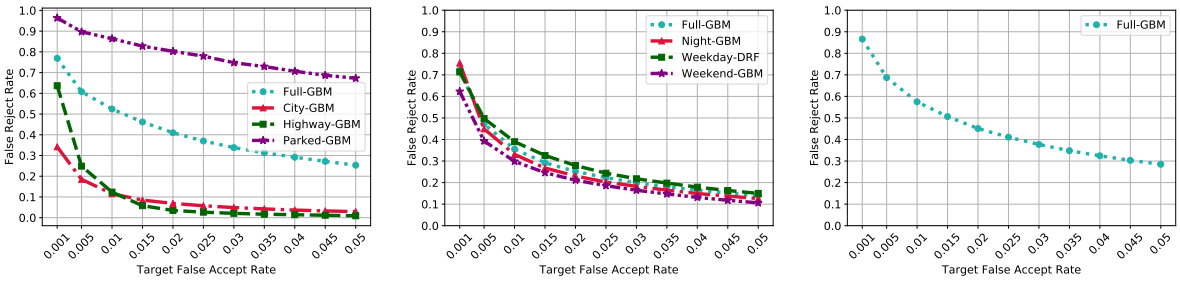
4.4.1 Methodology. To investigate the performance of machine learning colocation prediction, we use the H2O framework [33] to train a set of classifiers and pick the best performers. We evaluated Gradient Boosting Machines (GBMs) [10] and Random Forests (DRFs) [3] as classifiers, and then select the algorithm that gives the best cross-validated performance. These classifiers perform well in a wide range of datasets [6], they are fast, and they can handle instances with missing data directly in the model, allowing us to use instances with missing data in our datasets, which would otherwise have to be discarded. This is desirable, as in the real world, data may be incomplete (e.g., due to missing GPS fixes). These partial instances still provide information about the generating distribution and therefore are beneficial for the model, as shown by Tang *et al.* [32]. When building the cross-validation folds, H2O uses stratified sampling. This helps alleviate issues that can arise from class imbalances such as in datasets that contain more non-colocated than colocated instances.

To rank the classifiers, we use 10-fold cross-validation (CV) and estimate the Area Under the Curve (AUC), which measures the quality of the predictions irrespective of the selected thresholds. A higher AUC indicates a more accurately discriminative model. Using this measure is valid in this case, as we are interested in lower false accept and false reject errors along the predicting threshold domain.

For the learning, we let H2O split the data into training and validation datasets of 80% and 20% respectively. H2O will train a set of models independently from each other and automatically perform a parameter search to find optimal parameters for the specific dataset. Once we have found the top performing models, we get the cross-validated predictions $\hat{y} \in [0, 1]$. To convert those predictions to actual classes we use a threshold T and classify predictions that satisfy $\hat{y} > T$ as colocated. By optimizing the threshold, we balance the values between FAR and FRR to obtain the EERs or our target FARs. We also evaluate the impact of the individual features in the process using the normalized relative importance. The authors evaluated different interval length and came to the conclusion that increasing t above 10 seconds did not significantly increase the performance of the scheme. To validate this result, we evaluate two datasets, with $t = 10$ and 30. We present our results in Table 6 and 7.

4.4.2 Car. In this scenario, we obtain an EER of 0.111 and 0.104 for $t = 10$ and 30, respectively. Cross-correlation and time-frequency distance of the audio recordings account for 85% of the relative feature importance. This is expected, as the route passed through many areas without WiFi APs and BLE devices.

When investigating subscenarios, we observe that the parked subscenario exhibits a significantly higher EER than the other subscenarios. In this subscenario, the models also show a lower reliance on audio features, with those features making up only 64% of importance, and a higher precedence being given to WiFi and BLE features. This can likely be explained by the low audio activity in this subscenario, leading the model to use these less reliable features and thus reducing classification performance.



(a) Car (b) Office (c) Mob/het

Fig. 8. FRRs with target FARs for Truong *et al.* ($t = 10$)

The FRRs, given in Figure 8a, show similar trends: city and highway exhibit the lowest FRRs for the desired FARs, with the parked car significantly above them, and the full dataset somewhere in between. The FRRs also show a steeper drop in the beginning that tapers off later.

To test the robustness of the model, we use it to obtain predictions on the data from the other scenarios, applying the EER threshold determined before. This results in significantly increased error rates for all combinations of scenarios and intervals, with FARs larger than 0.44 for the office scenario, and FRRs in excess of 0.6 for the mob/het scenario, indicating that the model's performance will deteriorate when used on data from a scenario it has not been trained on and thus is not robust to being operated in different environments.

The robustness of models trained on subscenario datasets shows significant variation. Combinations of the city and highway subscenarios show changes between 0 and 4 percentage points, while combinations involving the parked subscenario show changes between 25 and 82 percentage points. This indicates that the models are robust to small changes in the environment, but cannot adapt to significant deviations.

4.4.3 Office. We observe a slightly improved EER of 0.084 ($t = 10$) and 0.069 ($t = 30$). Surprisingly, the WiFi features are not more relevant, despite WiFi being one of the best features reported by Truong *et al.*, and one would expect more stable signals for stationary devices compared to the mobile car scenario. However, our results show the audio features are even more relevant with a combined relative importance of 91%. To investigate if this is caused by the missing WiFi data in the dataset (cf. Section 5), we repeat the analysis, excluding instances where the WiFi data is missing due to a scan error, and obtain unchanged results. Thus, even in a dataset that contains WiFi data for all samples, the feature does not become more relevant for the classifier. The subscenarios show a much more similar behavior than in the previous scenario, with EERs between 0.071 (weekend) and 0.087 (weekday). This trend is also shown in the FRR evaluation in Figure 8b, where the curves are all closely matched.

When running the model on the car dataset for $t = 10$, we obtain an FAR and FRR of 0.174 and 0.412, respectively, with $t = 30$ increasing the error even further. Applying it to the mob/het dataset yields error rates of 0.022 and 0.711, respectively, once again increasing further for $t = 30$. This shows that the models are sufficiently different such that generalization is low and therefore robustness of the scheme suffers. Switching between the different subscenarios results in less pronounced changes, but still in some cases doubles the error rates. The scheme appears especially challenged when applying the weekend model to the other subscenarios, often doubling the error rates, while the weekday model is fairly robust, with only minor changes to most error rates. This is likely due to the higher complexity of the weekday dataset, which contains data from a more diverse set of situations.

4.4.4 Mob/het. In the mob/het scenario, we obtain EERs* of 0.127 and 0.123, respectively (cf. Table 8). Once again, the most important features are audio-based, although their importance is less pronounced, making up

only 60% and 56% of relative feature importance for $t = 10$ and 30, respectively. Optimizing for a low FAR will result in FRRs between 0.9 and 0.3 (cf. Figure 8c). This lower overall performance and the reduced prominence of the audio features is likely related to the issue of heterogeneous microphone characteristics, which we previously observed in the scheme proposed by Karapanos *et al.* (cf. Section 4.1.4), as Truong *et al.* use similar audio features.

Using the model to classify the car and office datasets results in significantly increased error rates (FAR > 0.7, FRR ≤ 0.22 for all combinations), showing that the model is not robust to different environments.

4.4.5 Conclusion. Our evaluation shows that the scheme can achieve a good EER in some of our scenarios, although it does not reach the error rates of the original paper, which observed a FAR and FRR of 0.0198 and 0.0167 for $t = 10$. We also see that models generated in one scenario show a significant loss in accuracy when being used in another scenario, and that the scheme encounters problems when using heterogeneous microphones. The authors also performed an experiment where pairs of devices were placed in close proximity (which matches our office scenario), obtaining a FAR of 0.0476, but did not report the FRR, which prevents a direct comparison.

Contrary to the original evaluation, the classification performance increased with larger intervals. We also saw a much higher importance of the audio feature than the original paper and a correspondingly lower importance of the WiFi feature. This is likely related to the collection strategy employed by the authors, who collected their dataset in different locations across two cities, which can be easily distinguished by their different WiFi signals.

The subscenario evaluation shows that the system does not work well in environments with little context activity, like cars parked in areas without WiFi and BLE devices. The differences between the subscenarios were less pronounced in the office scenario, where a larger number of WiFi and BLE devices were visible at all times.

Two factors limit the validity of our results. First, we did not collect GPS data, used by the authors. We assume that the impact would have been low in the office and mob/het scenarios, where devices were located close to each other and mostly static, however, it may have improved performance in the car scenario. Second, we use a different classifier than the authors, who used a Multiboost classifier [38], which is not supported in H2O. Still, DRFs and GBMs use ensemble methods similar to Multiboost and are unlikely to give significantly worse results.

4.5 Shrestha *et al.*

Shrestha *et al.* [28] propose combining readings from temperature, humidity, altitude, and precision gas sensors to decide if two devices are colocated. They compute the absolute difference between the readings of two devices and use a Multiboost classifier [38] trained on a labeled dataset to distinguish colocated and non-colocated devices. As our devices did not feature a precision gas sensor, we omit this feature. The sensor readings are not averaged over time intervals but used individually. Their dataset was obtained by collecting data from several locations using a pair of devices. Any data collected at different locations and times is interpreted as non-colocated. Additional details of the scheme are given in Section A.5.

4.5.1 Methodology. Although the machine learning methodology is identical to that used for the paper by Truong *et al.* (cf. Section 4.4 for details), the characteristics of the datasets and volume of data demand different treatment. One assumption made by any classifier in machine learning is that it estimates a surjective function from a vector of features $\hat{x}$ to a particular class c, i.e., all unique instances in the dataset map to exactly one class. However, our datasets do not fulfill this requirement, as several identical instances map from the same feature values to different classes. As the classifier has no additional data to base its decision on, it is unable to distinguish these ambiguous instances and thus can never reach a performance of 100%, i.e., the EER has a lower bound larger than 0. This indicates that more features are needed to discriminate the classes properly. We show the percentage of these ambiguous instances (Amb.) in each dataset in Table 9.

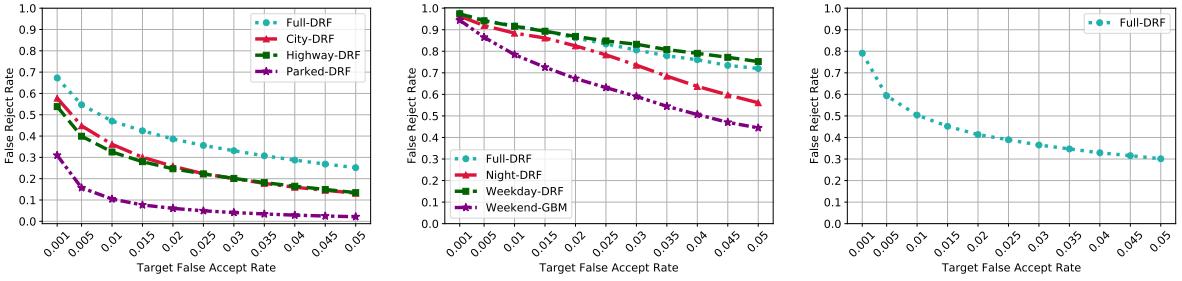
At the same time, it also indicates a potential for compression. Indeed, after analyzing the original office dataset with a size of 81 GB, we observe that many instances are repeated. Therefore, we introduce a pre-processing step

Table 8. Classification results for Truong *et al.*, Mob/het

Scenario	t	Model	EER	AUC	Acc.
Mob/het	10	GBM	0.127*	0.946	0.873%
Mob/het	30	GBM	0.123	0.949	0.877%

Table 9. Classification results for Shrestha *et al.*

Scenario	Model	EER	AUC	Acc.	Amb.
Car	DRF	0.115	0.960	88.5%	8.8%
– City	DRF	0.081*	0.977	91.9%	5.5%
– Highway	DRF	0.08	0.979	91.9%	6.8%
– Parked	DRF	0.034*	0.995	96.5%	2.2%
Office	DRF	0.247*	0.834	75.2%	25.5%
– Night	DRF	0.155*	0.911	84.4%	15.5%
– Weekday	DRF	0.271*	0.824	72.9%	25.5%
– Weekend	GBM	0.148*	0.928	85.1%	14.1%
Mob/het	DRF	0.141*	0.942	85.9%	12.2%



(a) Car (b) Office (c) Mob/het

Fig. 9. FRRs with target FARs for Shrestha *et al.*

before training, where we group all equal instances and keep a count of how many times they appear. These counts are used as weights in the later learning stage, which acts as a lossless compression mechanism. This way, we reduce the dataset to approximately 600 MB, which allows us to train models much faster and with significantly lower computational resources without sacrificing classification performance.

4.5.2 Car. For the car scenario, we obtain an EER* of 0.115, with the classifier relying almost evenly on all the features to make the predictions. The individual subscenarios achieve even lower EERs, showing rates of 0.034 (parked), 0.08 (highway) and 0.081 (city). This performance correlates with the percentage of ambiguous instances, with subscenarios with more ambiguous instances obtaining higher error rates. The low error rate of the parked subscenario is likely related to the use of temperature sensors, which captured the different rates of heat dissipation of the cars after they were parked. When aiming for a specific FAR, the differences between the subscenarios are maintained, with the parked subscenario showing consistently lower FRRs (cf. Figure 9a).

The model is not very robust, showing significantly increased error rates when applied to the office or mob/het dataset (FAR 0.342, FRR 0.503 for office, 0.276 / 0.717 for mob/het). Similarly, applying models specific to one subscenario to another increases the error rates by at least 32 percentage points.

4.5.3 Office. Here, the classifier reaches an EER* of 0.247, showing a significantly lower performance than in the car scenario. It relies strongly on the temperature differences to make the predictions. Such a focus on a

feature with a low range of potential values may make the classifier more vulnerable to active attacks and is thus undesirable. The low performance is mirrored in the subscenarios, with error rates of 0.148 (weekend) to 0.271 (weekday), which is also borne out in the high FRRs when aiming for a specific FAR (cf. Figure 9b). Once again, higher percentages of ambiguous instances translate to higher error rates. We also observe that the percentage of ambiguous instances grows with the size of the dataset. This is to be expected, as a larger dataset has a higher chance of obtaining these instances, as the range of potential values is limited.

When the office model is used on the other two datasets, the error rates increase significantly (FAR 0.186, FRR 0.693 for car, 0.184 / 0.787 for mob/het), showing that the model is not robust to being used in different environments. Of the three subscenario models, the weekday model is the most robust, with error rate increases of below 6 percentage points when applied to different subscenario datasets. However, applying the night model to weekday data increases the error rate by 35 percentage points, and the weekend model increases its error rate by over 43 percentage points with the weekday dataset. This shows that the robustness is limited.

4.5.4 Mob/het. Due to a lack of humidity and temperature sensors, the laptops and robot were not included in the evaluation. The classifier reaches an EER* of 0.141 (cf. Table 9), relying primarily on the altitude readings, with a lower importance given to temperature and humidity. A closer investigation revealed that the phones' barometric pressure readings deviated from those of the watches and SensorTags, showing an offset of approximately 2 hPa. The temperature and humidity readings varied widely, with the position of the device (smartphone in pocket, sensor on screen, ...) having a much larger influence than the room they operate in. The error rate is likely related to this challenging environment, as well as the number of ambiguous instances, which make up 12.2% of the dataset. When optimizing for a low FAR, the resulting FRR are at least 0.3 (cf. Figure 9c).

Applying the model to the car dataset results in notably increased error rates (FAR 0.302, FRR 0.672). The office dataset gives similar error rates (0.306 / 0.606), showing limited robustness of the model in different environments.

4.5.5 Conclusion. Overall, the scheme by Shrestha *et al.* cannot reliably separate colocated from non-colocated devices in most scenarios. This is in stark contrast to the error rates reported by the authors, who obtained an FAR and FRR of 0.0581 and 0.0296, respectively. This deviation can partially be explained by the lack of precision gas features in our datasets, reducing the number of dimensions the models can discriminate on. Another explanation is the more challenging environment our data was collected in—the authors collected their data in widely spaced locations at different times of day, and their non-colocated class consisted of pairings between different locations. This also explains the high discriminative power of the altitude readings reported by the authors, and indicates that the scheme will likely have a much better performance if only coarse colocation is required.

The high number of ambiguous instances shows that the scheme would benefit from incorporating additional sensors to improve its discriminative power. Additionally, features tracking the change in values over time may be a more promising approach, as our dataset shows these changes to be more consistent between devices in the same room than the sensor readings themselves. The results also show that even if high classification performance can be reached, it is still highly specific to the environment it was collected in, and does not transfer well into other environments, i.e., the robustness of the scheme is limited.

5 DISCUSSION

In this section, we discuss the implications of the obtained results, the limitations of our method, and avenues for future work.

Performance Comparison. Our results, summarized in Table 2, show that the scheme by Truong *et al.* obtains the best EERs in the office (0.069) and mob/het (0.123) scenarios, while the scheme by Karapanos *et al.* achieves a significantly lower error rate (0.006) in the car scenario. This indicates that depending on the use case, both are solid choices. We also observed a large variation in performance on different subscenarios, ranging from perfect

accuracy (Karapanos *et al.*, $t = 120$, highway) to significantly degraded performance compared to using the full dataset (Schürmann *et al.*, $t = 120$, parked), illustrating the importance of fine-grained test scenarios.

Adaptiveness. Some schemes struggle to adapt to times of low ambient activity like the night, where we observed a lower separation between colocated and non-colocated devices. These times need to be taken into account when designing a scheme intended for continuous operation. Karapanos *et al.* and Miettinen *et al.* added measures to reduce the impact of these times by dynamically discarding samples with low ambient activity [14] or high predictability [20], trading off availability for security.

Robustness. Even if they can operate in environments with low activity, most schemes suffer from a lack of robustness, i.e., parameters that are optimal for one scenario do not give good performance on the other. Some schemes can achieve a certain degree of robustness for specific parameters (e.g., interval sizes), most notably that by Karapanos *et al.*, but no scheme is robust with all parameters. The same trend holds when exchanging parameters or models between different subscenarios, like day and night, especially if they have significantly different ambient activity levels. This further illustrates the importance of testing schemes in a wide variety of settings. We urge researchers to pay special attention to robustness, to facilitate use in the wide variety of different and sometimes unexpected environments the IoT will be deployed in.

Heterogeneity. Even if schemes can provide good performance in settings with homogeneous devices (i.e., the same hardware), they may still fail when encountering devices with different characteristics. These challenges have also been encountered in other research fields such as participatory sensing. Examples from our dataset include microphones with varying sensitivity and frequency response [15, 17, 18], which leads to lower correlations, or incorrectly calibrated sensors (e.g., air pressure) measuring with a fixed offset from one another. In addition, the way a device is carried influences the observed sensor data [21]. Schemes need to be able to still provide good results under these conditions if they are intended for use cases where the used hardware is not carefully controlled by a single party and should be tested with heterogeneous devices.

Colocation definition. Many schemes do not explicitly state their colocation definition. It is often unclear if they are intended to distinguish personal workspaces inside an office, different rooms, or parts of a city, making it difficult to identify security guarantees schemes provide in any specific situation. This hinders a fair evaluation and comparison of these schemes, and makes it hard to determine if our results impact its designated use case. Authors should explicitly define what their scheme considers (non)colocated to allow for fair comparisons.

Limitations. Technical issues during the recording led to data loss for some features, especially the WiFi captures, which stopped working on some devices. In the mob/het scenario, three devices stopped the data collection before the eight-hour countdown, resulting in partial loss of audio and sensor data. This reduces the amount of available data for the evaluation of features based on these modalities. In the same way, the *SensorTag* platform occasionally delivered incorrect readings for the luminosity, which we detected and excluded.

Our goal was to compare the different ZIS schemes in a fair manner and as specified by their original authors. While we attempted to stay as close to the published version of the scheme as possible, in some cases, minor changes had to be made to parameters such as interval length or sampling rate. These deviations are noted in the Appendix, and their influence on the results should be negligible. We did not attempt to optimize any parameters aside from interval length and the power threshold of Karapanos *et al.* for the mob/het scenario for our dataset, so it is possible that some schemes could perform better when they are instantiated with different parameters.

We also note that our scenarios are challenging, as they include devices in isolated positions (glove compartment, cupboard, pocket), low isolation between two offices, and two cars that traveled next to each other for extended amounts of time. This is intentional to be able to investigate the performance of the systems in challenging situations, as consumer IoT deployments rarely follow best practices for a deployment facilitating zero-interaction

security. We also did not include any scenarios in busy areas like shopping malls, which may show different behavior due to the higher environmental variations, as we chose to focus on the likely application domain of zero-interaction technologies, consumer applications. Additionally, obtaining approval and informed consent for a long-term data collection in a public place would have been infeasible in our jurisdiction.

Future Work. Our chosen scenarios only cover a subset of interesting IoT environments. Other scenarios may pose different challenges for the schemes. For example, in a *smart building* scenario, an ideal ZIS scheme would have to be able to authenticate all devices within the same building, while excluding adjacent buildings. In a *café* scenario, schemes would need to be able to distinguish individual tables. We also did not include any scenarios where devices operated in environments without any humans for extended amounts of time (e.g., automated factory work floors or storage units), which could pose challenges to many schemes due to the potentially low variation in the context information or different noise characteristics.

The collection of additional datasets will assist efforts to create more adaptive and robust schemes and to understand the limitations of existing ones. Another avenue for future work is the robustness to adversarial settings, where part of the context information can be controlled by an active adversary (e.g., by injecting sound).

6 CONCLUSION

We reproduced and evaluated five ZIP and ZIA schemes in three realistic scenarios—*smart office*, *connected car* and *smart office with mobile heterogeneous devices*—posing different challenges in aspects like environmental noise, context leakage, and times of low activity. Our results show that none of the reproduced schemes can perfectly separate devices in all scenarios. The schemes by Karapanos *et al.* [14] and Truong *et al.* [36] show promising results, but no scheme reliably outperforms all others in all scenarios. The error rates also indicate that zero-interaction security should not be used as the only access control factor, as even a false accept rate of 1% would be considered insufficient for some real-world applications. In fact, Karapanos *et al.* explicitly proposed their ZIA scheme as a more convenient second authentication factor [14] instead of a stand-alone solution.

We also observed that a good average-case separation of context features aggregated over the whole dataset does not imply a good authentication performance on individual samples. ZIP and ZIA schemes should thus be evaluated in terms of their error rates in both the average case as well as individual subscenarios to get a realistic impression of their performance. In the same way, the evaluation should be performed using a set of heterogeneous devices in a realistic, challenging environment, to test the limits of the scheme.

Our evaluation revealed that in many cases, features based on ambient audio performed best. However, researchers need to take the privacy implications of using audio recordings into account, as this may not be acceptable in some environments like hospitals. Additionally, the computational costs of processing have to be considered, as expensive audio processing operations may not always be possible on resource-constrained IoT devices. Finally, we observed that devices with differing microphone characteristics can significantly degrade the performance of sound-based schemes. Thus, we encourage researchers to continue to investigate the possibility of using more power-efficient features based on low-power sensors. Here, we found that instead of using the absolute difference between sensor readings, trends over time may be a more reliable colocation indicator.

We also observed that the robustness and adaptiveness of many schemes varies dramatically for different scenarios. Schemes should explicitly state which environments they are designed for. Additionally, they should support robustness and adaptiveness, potentially by automatically adapting their internal parameters to their environment, and should be evaluated on data from different scenarios and devices.

Finally, we release the first extensible open source toolkit [9] for researching zero-interaction security, containing reference implementations of the reproduced schemes, the audio recordings of the *mob/het* scenario, and over 1 billion samples of labeled sensor data. We also release all data generated by our evaluation to facilitate the reproduction of our results and provide a common benchmarking baseline for future schemes.

ACKNOWLEDGMENTS

We would like to thank Daniel Wegemer, Jiska Classen, Timm Lippert, Robin Klose, Vanessa Hahn and Santiago Aragón for assistance in conducting this research. Furthermore, we are thankful to Hien Truong, Nikolaos Karapanos, Dominik Schürmann, Markus Miettinen and Babins Shrestha for assistance in reproducing their work. This work has been co-funded by the DFG within CRC 1119 CROSSING and CRC 1053 MAKI projects, and as part of project C.1 within the RTG 2050 "Privacy and Trust for Mobile Users". Calculations for this research were conducted on the Lichtenberg high performance computer of the TU Darmstadt.

APPENDIX

A REPRODUCED ZIS SCHEMES

In this appendix, we provide more details about the reproduced ZIS schemes. We give a brief overview of each scheme's functionality and use case and describe the implementation of context features utilized by the scheme.

Audio preprocessing: Before computing audio features we aligned audio recordings from different sensing devices as follows. At the beginning of data collection all devices were synchronized using the Network Time Protocol (NTP). First, we performed the coarse-grained alignment using devices' timestamps to synchronize the start of the audio recordings. Second, during the feature computation we performed a fine-grained alignment between two input audio recordings using the cross-correlation function in Matlab [35]. Specifically, we considered the first hour of audio recordings to find a lag between them, using the *xcorr* function ($maxlag = 3$ seconds), then we used this lag to align two audio recordings and cut them to the length of the shortest recording. These aligned recordings are then split into intervals and used to compute audio features.

In the mobile scenario, we increased the *maxlag* to 15 seconds to more precisely find the lag between audio recordings of heterogeneous devices. In addition, we found that heterogeneous devices have an inherent audio drift, causing desynchronization of audio recordings. We removed this drift by applying a time-stretching effect to audio recordings in the *Audacity* tool (change Tempo).

A.1 Karapanos *et al.*

The scheme by Karapanos *et al.* [14] calculates a similarity score between snippets of ambient audio from two devices to decide if these devices are colocated. The similarity score is the average of the maximum cross-correlations between two audio snippets computed on a set of one-third octave bands. To prevent erroneous authentication when audio activity is low, a power threshold is applied to discard similarity scores from audio snippets with insufficient average power. The similarity score is then checked against a fixed similarity threshold to decide if two devices are colocated. The scheme is designed to provide colocation evidence between a user's smartphone and a computer with a running browser. This evidence is utilized as a second authentication factor when a user wants to log-in to an online service such as a bank account. In this work, we focus on computing and comparing similarity scores and do not target the specific use case of the second authentication factor.

We first provide notations adopted from the original paper in Table 10. Second, we present parameters of the sound similarity algorithm used in the original and our implementations in Table 11. Our goal was to follow the original implementation as close as possible, however, we introduced a few changes, as we did not have tight synchronization between audio snippets. Third, we present our implementation of the sound similarity algorithm in Section A.1.1.

As shown in Table 11, our implementation differs with respect to these parameters from the implementation by Karapanos *et al.* First, we increase both the length of input audio snippets L from 3 to 5 seconds and the length of the maximum cross-correlation lag l_{max} from 0.15 to 1 second to achieve a comparable level of authorization to the authors. We observed that even after the alignment procedure (cf. *Audio preprocessing*) there might be an

Table 10. Notations used by Karapanos *et al.*

Notation	Explanation
x, y	input audio snippets
L	length of input audio snippets in seconds
l_{max}	max cross-correlation lag in seconds
r	sampling rate of input audio snippets in kHz
τ_{dB}	average power threshold in dB
B	set of considered one-third octave bands
n	number of considered one-third octave bands
$S_{x,y}$	similarity score

Table 11. Parameters of the sound similarity algorithm used in [14] (highlighted) and our implementations

L, sec	l_{max}, sec	r, kHz	τ_{dB}, dB	B (n)
3	0.15	44.1	40	50Hz – 4kHz (20)
5	1	16	40/38/35	50Hz – 4kHz (20)
10	1	16	40/38/35	50Hz – 4kHz (20)
15	1	16	40/38/35	50Hz – 4kHz (20)
30	1	16	40/38/35	50Hz – 4kHz (20)
60	1	16	40/38/35	50Hz – 4kHz (20)
120	1	16	40/38/35	50Hz – 4kHz (20)

offset within long audio recordings (24 hours), which can affect synchronization between audio snippets. That is why, we set $l_{max} = 1$ to maintain a balance between security (l_{max} thwarts attackers trying to guess the audio environment) and non-tight synchronization, which can happen in a realistic IoT scenario. The increase of l_{max} leads to the increase of the audio snippet length L to 5 seconds. Second, we use a lower sampling rate for the input audio snippets r: 16 vs. 44.1 kHz, which does not affect the sound similarity algorithm itself, but can be used to speed up the computations, as a smaller number of samples needs to be processed. Despite the lower sampling rate and, thus, narrower audio spectrum (8 kHz) we cover the same set of octave bands as the original implementation.

As stated in Section 4, we evaluate the performance of the scheme on a number of intervals from 5 to 120 seconds.

A.1.1 Implementation of the sound similarity algorithm.

(0) As input, we have two aligned audio snippets x and y of equal length L with a sampling rate r.
(1) Both x and y are split into n one-third octave bands using a bank of band-pass filters:

$$\begin{aligned}(x_{B_1}, \ldots, x_{B_n}) &= \text{BP\_filter\_bank}(x)\\ (y_{B_1}, \ldots, y_{B_n}) &= \text{BP\_filter\_bank}(y)\end{aligned} \tag{1}$$

Table 12 shows the used one-third octave bands B from 50 Hz to 4 kHz, and each band-pass filter is constructed as a 20th-order Butterworth filter [34] with cut-off frequencies $[F_l, F_h]$.

(2) For each x_{B_i} and $y_{B_i} \forall i \in [1, n]$ the normalized maximum cross-correlation $\hat{C}_{x,y}(l)$ is computed as the function of the lag $l \in [0, l_{max}]$ (we omit B_i indexes for simplicity):

$$\hat{C}_{x,y}(l) = \max_l(|C'_{x,y}(l)|) = \max_l\left(\left|\frac{C_{x,y}(l)}{\sqrt{C_{x,x}(0) \cdot C_{y,y}(0)}}\right|\right) \tag{2}$$

In Equation 2 the term $C_{x,y}(l)$ is a cross-correlation function between two discrete signals x and y:

$$C_{x,y}(l) = \sum_{i=0}^{N-1} x(i) \cdot y(i-l) \tag{3}$$

N is the number of samples in the signals[1] and the lag is bounded within a range $l \in [0, N-1]$. The normalization term $\sqrt{C_{x,x}(0) \cdot C_{y,y}(0)}$ accounts for different amplitudes of signals x and y, with $C_{x,x}(0)$ and $C_{y,y}(0)$ being the auto-correlation functions. The resulting maximum cross-correlation is bounded within a range $\hat{C}_{x,y}(l) \in [0, 1]$, because we take the absolute value of the normalized cross-correlation $|C'_{x,y}(l)|$.

(3) The resulting similarity score between two audio snippets x and y is obtained by taking the average of the normalized maximum cross-correlations computed in each one-third octave band:

$$S_{x,y} = \frac{1}{n}\sum_{i=1}^{n} \hat{C}_{x_{B_i}, y_{B_i}}(l) \tag{4}$$

The similarity score is only used if the input audio snippets have sufficient average power: $\overline{P}_x, \overline{P}_y > \tau_{dB}$. Otherwise, it is discarded and no authentication is attempted.

A.2 Schürmann and Sigg

The scheme by Schürmann and Sigg [24] computes a binary fingerprint from a snippet of ambient audio, based on energy differences in successive frequency bands. Two devices wishing to establish a pairing compute such fingerprints from their ambient environments. These fingerprints are used in a fuzzy commitment scheme to obtain a shared secret. One device uses its fingerprint to hide a randomly chosen secret and sends this commitment to the other device, which can only retrieve the random secret from the commitment if it has a sufficiently similar fingerprint. In this work, we focus on deriving and comparing binary fingerprints and we do not target a specific use case of establishing a shared secret key.

We first provide notations adopted from the original paper in Table 13. Second, we present parameters of the audio fingerprinting algorithm used in the original and our implementations in Table 14, where we introduce a few changes, as our audio snippets have a lower sampling rate. Third, we present our implementation of the audio fingerprinting algorithm in Section A.2.1.

As shown in Table 14, our implementation differs with respect to some parameters from the implementation by Schürmann and Sigg. First, we use a lower sampling rate r of 16 kHz instead of the original 44.1 kHz, which affects the number of frequency bands m we can split our frames n into. With a 16 kHz sampling rate our audio spectrum is only 8 kHz, thus we can only obtain 32 non-overlapping frequency bands, each of width 250 Hz b. Having 32 frequency bands instead of 33 as in the original implementation results in shorter binary fingerprints f of 496 instead of 512 bits. Second, we vary the lengths l from 5 to 120 seconds, which also affects the length of a single frame d, which varies between 0.29 and 7.06 seconds. We note that shorter audio frames (e.g. $d = 0.29$) are more susceptible to synchronization issues between input audio snippets, thus reducing the similarity of binary fingerprints generated from these snippets. However, starting from $l = 10$ our frame length d is bigger than in

[1]We assume signals x and y have the same length

Table 12. Used one-third octave bands

Band Number	F_l, Hz	F_c, Hz	F_h, Hz
6	44.194	49.606 (50)	55.681
7	55.681	62.500 (63)	70.154
8	70.154	78.745 (80)	88.388
9	88.388	99.213 (100)	111.362
10	111.362	125.000 (125)	140.308
11	140.308	157.490 (160)	176.777
12	176.777	198.425 (200)	222.725
13	222.725	250.000 (250)	280.616
14	280.616	314.980 (315)	353.553
15	353.553	396.850 (400)	445.449
16	445.449	500.000 (500)	561.231
17	561.231	629.961 (630)	707.107
18	707.107	793.701 (800)	890.899
19	890.899	1000.000 (1000)	1122.462
20	1122.462	1259.921 (1250)	1414.214
21	1414.214	1587.401 (1600)	1781.797
22	1781.797	2000.000 (2000)	2244.924
23	2244.924	2519.842 (2500)	2828.427
24	2828.427	3174.802 (3150)	3563.595
25	3563.595	4000.000 (4000)	4489.848

F_l - lower band frequency, Fc - calculated center frequency (nominal frequency), F_h - upper band frequency

Table 13. Notations used by Schürmann and Sigg

Notation	Explanation
S	input audio snippet
l	length of the input audio snippet in seconds
r	sampling rate of the input audio snippet in kHz
n	number of frames to split the input audio snippet
m	number of frequency bands to split each frame
d	length of each frame in seconds (duration)
b	width of each frequency band in Hz
f	binary fingerprint of length $(n-1)\cdot(m-1)$ in bits

the original implementation, which makes our results comparable and allows us to access the performance of the scheme (i.e. distinguishing between colocated and non-colocated devices) on longer audio snippets.

A.2.1 Implementation of the audio fingerprinting algorithm.

(0) As input, we have an audio snippet S of length l with a sampling rate r (audio snippets from different devices are aligned). The number of frames n and the number of frequency bands m are selected to obtain

Table 14. Parameters of the audio fingerprinting algorithm used in [24] (highlighted) and our implementations

f, bits	l, sec	r, kHz	n, frames	m, bands	d, sec	b, Hz
512	6.375	44.1	17	33	0.375	250
496	5	16	17	32	~0.29	250
496	10	16	17	32	~0.59	250
496	15	16	17	32	~0.88	250
496	30	16	17	32	~1.76	250
496	60	16	17	32	~3.53	250
496	120	16	17	32	~7.06	250

the binary fingerprint of the desired length:

$$L_f = (n-1) \cdot (m-1) \tag{1}$$

The width of a frequency band depends not only on the number of bands but also on the available audio spectrum which is limited by the Nyquist frequency ($f_N = \frac{r}{2}$):

$$b = \frac{\text{maxfreq}(S) - \text{minfreq}(S)}{m} \tag{2}$$

(1) The audio snippet S is split into n successive frames $F_1, \ldots, F_n$ of equal length $d = r \cdot \frac{l}{n}$ in samples (F_i is a dx1 vector).

(2) Each frame $F_1, \ldots, F_n$ is split into m non-overlapping frequency bands of width b using a bank of band-pass filters:

$$(F_{B_1}, \ldots, F_{B_m})_i = \text{BP\_filter\_bank}(F_i), \quad \forall i \in [1, n] \tag{3}$$

In our implementation the available audio spectrum is 8 kHz, thus we split it into the following 32 bands of width 250 Hz: $B_1 = [1, 250], B_2 = [251, 500], \ldots, B_m = [7751, 7999]$, using a 20th-order Butterworth filter [34] for each band.

(3) For each frame $F_1, \ldots, F_n$ the energy of each frequency band $B_1, \ldots, B_m$ is computed as (superscript T denotes transpose):

$$(E_{B_j})_i = (F_{B_j}^T \cdot F_{B_j})_i, \quad \forall i \in [1, n]; \forall j \in [1, m] \tag{4}$$

(4) The results of energy computation are stored in the energy matrix ($\forall i \in [1, n]; \forall j \in [1, m]$):

$$E_{i,j} = \begin{pmatrix} E_{F_1,B_1} & E_{F_1,B_2} & \cdots & E_{F_1,B_m} \\ E_{F_2,B_1} & E_{F_2,B_2} & \cdots & E_{F_2,B_m} \\ \vdots & \vdots & \ddots & \vdots \\ E_{F_n,B_1} & E_{F_n,B_2} & \cdots & E_{F_n,B_m} \end{pmatrix} \tag{5}$$

(5) The binary fingerprint f is obtained by iterating over consecutive frames $\forall i \in [1, n-1]$ and frequency bands $\forall j \in [1, m-1]$. Each bit of the fingerprint is generated by checking the energy difference between successive frequency bands of two consecutive frames ($\forall k \in [1, L_f]$):

$$f_k = \begin{cases} 1, & (E_{i+1,j} - E_{i+1,j+1}) - (E_{i,j} - E_{i,j+1}) > 0 \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

A.3 Miettinen *et al.*

The scheme by Miettinen *et al.* [20] is inspired by the audio fingerprinting scheme proposed by Schürmann and Sigg (cf. A.2) but works on longer timescales. It uses noise level and luminosity measurements to derive long-term binary fingerprints, which can defend against adversaries that are colocated for short timeframes. The scheme utilizes such fingerprints in a fuzzy commitment scheme (as described in A.2) to gradually evolve a shared secret key to achieve pairing between two devices that are colocated for a sustained period of time. In this work, we focus on deriving and comparing long-term binary fingerprints and we do not target a specific use case of establishing a shared secret key.

We first provide notations adopted from the original paper in Table 15. Second, we present parameters of the context fingerprinting algorithm used in the original and our implementations in Table 16. Our goal was to follow the original implementation as close as possible, however, we introduced a few changes, as we use audio with a higher sampling rate to generate noise levels. We discuss the effect of those changes on the parameters of the context fingerprinting algorithm. Third, we present our implementation of the context fingerprinting algorithm in Section A.3.1.

Table 15. Notations used by Miettinen *et al.*

Notation	Explanation
w	length of the context snapshot in seconds
f	a new snapshot is recorded every f seconds
r	sampling rate of recorded audio in kHz
m_w	measurement window in seconds
Δ_{rel}	relative threshold for fingerprint generation
Δ_{abs}	absolute threshold for fingerprint generation

Table 16. Parameters of the context fingerprinting algorithm used in [20] (highlighted) and our implementations

w, sec	f, sec	r, kHz	m_w, sec	Δ_{rel}	Δ_{abs}
120	120	8	0.1/1	0.1	10
5	5	16	1	0.1	10
10	10	16	1	0.1	10
15	15	16	1	0.1	10
30	30	16	1	0.1	10
60	60	16	1	0.1	10
120	120	16	1	0.1	10

As shown in Table 16, our implementation differs with respect to these parameters from the implementation by Miettinen *et al.* [20]. First, we use audio with a higher sampling rate r: 16 vs. 8 kHz to generate noise levels. The noise levels are generated by averaging absolute amplitudes of audio samples over m_w seconds, given by the measurement window. Thus, for $m_w = 1$, we obtain one noise level from 16000 audio samples, whereas the original implementation computes one noise level from only 8000 audio samples, which makes our noise levels more fine-grained. The original implementation uses two different measurement windows m_w: 0.1 and 1 sec. The shorter measurement window speeds up the fingerprint generation but may be susceptible to synchronization issues,

thus we opt for a longer measurement window. For luminosity measurements we do not use the measurement window. We collect luminosity readings at 10 samples per second and use all samples generated during context snapshot length w to obtain the fingerprint. Second, we evaluate the context fingerprinting algorithm on the context snapshots of different lengths w from 5 to 120 seconds. Thus, we can assess the performance of the scheme (i.e. distinguishing between colocated and non-colocted devices) on shorter context snapshots.

A.3.1 Implementation of the context fingerprinting algorithm.

(0) As input we have sets of noise level S_{nl} and luminosity S_{lux} measurements generated from context information collected in our scenarios (i.e. car and office) as stated above. The number of bits b in the resulting context fingerprints is given by $\frac{|S_{nl}|}{f}$ and $\frac{|S_{lux}|}{f}$, where $|\cdot|$ denotes the set cardinality.

(1) The *context snapshot* c_w for a timeslot t consists of all measurements m taken in the timeslot of w seconds, $c_w(t) = (m_i, m_{i+1}, \ldots, m_{i+n})$. For each context fingerprint the average value $\overline{c}(t)$ is computed as:

$$\overline{c}(t) = \frac{\sum\limits_{m_i \in c(t)} m_i}{|\{m_i \in c(t)\}|} \tag{1}$$

(2) Each set of measurements (S_{nl} or S_{lux}) can be represented as a sequence of context snapshots $C(t, t+nf) = (c(t), c(t+f), \ldots, c(t+nf))$. Then the fingerprint bit $b(t_i)$ which corresponds to each snapshot $c(t_i)$ is generated as:

$$b(t_i) = \begin{cases} 1, & \left|\frac{\overline{c}(t_i)}{\overline{c}(t_i - f)} - 1\right| > \Delta_{rel} \ \wedge \ \left|\overline{c}(t_i) - \overline{c}(t_i - f)\right| > \Delta_{abs} \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

We note that the values for Δ_{rel} and Δ_{abs} (cf. Table 16) are not given in the original paper but were provided by the authors in private communication.

(3) The resulting fingerprint for the set of measurements (S_{nl} or S_{lux}) is obtained as:

$$\phi(C(t, t+nf)) = (b(t), b(t+f), \ldots, b(t+nf)) \tag{3}$$

To avoid using fingerprints that are exclusively zero in times of low ambient noise and light, Miettinen *et al.* proposed an extension to their system: they propose to compute the *surprisal* of a fingerprint before using it. The surprisal of a single bit b of the fingerprint is defined as its self-information I, measured in bits:

$$\sigma(b) = I(b) = -log_2(P(B = b)) \tag{4}$$

The surprisal of the whole fingerprint F is the sum of the surprisal of its individual bits:

$$\sigma(F) = \sum_{b \in F} \sigma(b) \tag{5}$$

Calculating this surprisal requires knowledge about how often bits occur at specific positions of the fingerprint during specific times of the day, indicated as $P(B = b)$ in the formula. Miettinen *et al.* do not state the time resolution, but it is implied that the probabilities are tracked on a per-hour basis. For the office scenario, which covers multiple days, we track the probabilities independently for the individual days, i.e., fingerprints generated on weekdays do not influence the probabilities and thus surprisals for the weekend.

Miettinen *et al.* propose to set a surprisal threshold σ_{thr} that the surprisal of a fingerprint has to exceed in order to be considered valid for pairing. This avoids the problem of attacks by adversaries guessing the low-entropy fingerprints generated at night. The threshold is defined as

$$\sigma_{thr} = t + \sigma_{marg} \tag{6}$$

t denotes the number of incorrect bits the fuzzy commitment will tolerate and σ_{marg} denotes an extra security margin. However, the authors do not state how this margin should be chosen. Our margin choice is described in Section 4.3

A.4 Truong *et al.*

The scheme by Truong *et al.* [36] uses WiFi, Bluetooth, GPS and ambient audio collected by two devices to compute a number of context features, which are then fed into a machine learning classifier that outputs a prediction if these devices are colocated. This scheme is designed to provide colocation evidence to thwart relay attacks on wireless channels between a user's device and terminal, which employ ZIA (e.g. unlock a computer if a user's smartphone is nearby). In this work, we focus on computing context features and obtaining classification results from the machine learning algorithms and we do not target the specific use case of thwarting relay attacks.

We first provide notations adopted from the original paper in Table 17. Second, we describe how different context features are computed. Third, we provide details of our machine learning methodology, where we discuss our datasets, the parameters of machine learning algorithms that we use and the evaluation procedure.

Due to a lack of GPS support in the used hardware, we were unable to collect GPS information. However, since our office scenario is static and the car scenario mostly considers geographically close cars, the information value of the GPS features would have been low. In addition, the original authors report that the GPS feature contains the least amount of discriminative power in their dataset.

Table 17. Notations used by Truong *et al.*

Notation	Explanation
$m_i^{(a)}$	identifier of the ith beacon observed by device a
$s_i^{(a)}$	signal strength of ith beacon observed by device b
θ	value substituted for missing signal strengths
S_a	set of records sensed by device a
n_a	number of different beacons observed by device a
$S_\cap$	beacons seen by a and b
$S_\cup$	beacons seen by a or b, θ substituted for missing s
x, y	input audio snippets
L	length of input audio snippets in seconds
r	sampling rate of input audio snippets in kHz

A.4.1 Non-audio features. The features for WiFi, Bluetooth and GPS are defined over a number of sets. Individual samples for each context information are defined as a tuple (m, s), where m denotes the identifier of the observed beacon (i.e., BLE MAC address, WiFi BSSID) and s denotes the received signal strength. The set of records observed by devices a and b is denoted as S_a and S_b, respectively, while n_a and n_b denote the number of unique beacons observed by the devices. The notation is also given in Table 17. Given these preconditions, the following sets are defined:

$$
\begin{aligned}
S_a &= \{(m_i^{(a)}, s_i^{(a)}) \mid i \in \mathbb{Z}_{n_a-1}\} \\
S_b &= \{(m_i^{(b)}, s_i^{(b)}) \mid i \in \mathbb{Z}_{n_b-1}\} \\
S_a^{(m)} &= \{m \; \forall (m,s) \in S_a\} \\
S_b^{(m)} &= \{m \; \forall (m,s) \in S_b\} \\
S_\cap &= \{(m, s^{(a)}, s^{(b)}) \; \forall m \mid (m, s^{(a)}) \in S_a, \; (m, s^{(b)}) \in S_b\} \\
S_\cup &= S_\cap \; \cup \; \{(m, s^{(a)}, \theta) \; \forall m \mid (m, s^{(a)}) \in S_a, \; m \notin S_b^{(m)}\} \\
&\quad \cup \{(m, \theta, s^{(b)}) \; \forall m \mid (m, s^{(b)}) \in S_b, \; m \notin S_a^{(m)}\} \\
S_\cap^{(m)} &= \{m \; \forall m \mid (m, s^{(a)}, s^{(b)}) \in S_\cap\} \\
S_\cup^{(m)} &= \{m \; \forall m \mid (m, s^{(a)}, s^{(b)}) \in S_\cup\} \\
L_a^{(s)} &= \{s^a \mid (m, s^{(a)}, s^{(b)}) \in S_\cap\} \\
L_b^{(s)} &= \{s^b \mid (m, s^{(a)}, s^{(b)}) \in S_\cap\}
\end{aligned}
$$

Truong *et al.* uses these sets to define a total of six features, five of which we implement: the Jaccard Distance $J_{a,b}$, mean Hamming distance $\overline{H}_{a,b}$, Euclidean distance $E_{a,b}$, mean exponential of difference $\overline{Exp}_{a,b}$, the sum of squared ranks $S_{a,b}^{(sr)}$. The sixth feature, subset count, is only used for the GPS data and thus omitted. The features are given by the following formulas, where θ is specific to certain context information.

$$J_{a,b} = 1 - \frac{|S_\cap^{(m)}|}{|S_\cup^{(m)}|} \tag{1}$$

$$\overline{H}_{a,b} = \frac{\sum_{k=1}^{|S_\cup|} \left| s_k^{(a)} - s_k^{(b)} \right|}{|S_\cup|} \tag{2}$$

$$E_{a,b} = \sqrt{\sum_{k=1}^{|S_\cup|} (s_k^{(a)} - s_k^{(b)})^2} \tag{3}$$

$$\overline{Exp}_{a,b} = \frac{\sum_{k=1}^{|S_\cup|} \exp \left| s_k^{(a)} - s_k^{(b)} \right|}{|S_\cup|} \tag{4}$$

$$S_{a,b}^{(sr)} = \sum_{k=1}^{|S_\cap|} (r_k^{(a)} - r_k^{(b)})^2 \tag{5}$$

$|\cdot|$ denotes the set cardinality; $r_k^{(a)}$ ($r_k^{(b)}$) is the rank of $s_k^{(a)}$ ($s_k^{(b)}$) in the set L_a (L_b) sorted in ascending order.

For WiFi, all features are used. The signal strength s for each observed identifier is set to the average observed signal strength for that identifier over all included scans. θ, which is substituted as signal strength for devices

that have been observed by one but not the other device, is set to -100. For BLE, features 1 and 3 are used, once again using the average observed signal strength for each identifier as s and $\theta = -100$.

In case both sensors observe no beacons, the distances are not defined, and the original paper does not specify a behavior for this case. In private communication, the authors recommended choosing either zero (if the system should be biased towards accepting) or a very high number (if it should be biased towards rejecting). In our case, we chose to replace undefined values with the distance 10 000 to bias the system towards rejecting when in doubt.

A.4.2 Audio features. Truong *et al.* use two audio features: the maximum cross-correlation and time-frequency distance computed on snippets of ambient audio of length $L = 10$ seconds. The authors do not provide the sampling rate r of their audio snippets; in our implementation $r = 16$ kHz. We compute these context features on audio snippets of different lengths L from 5 to 120 seconds. In the end, we create and evaluate two different datasets for machine learning, one using $L = 10$, the other $L = 30$.

In the following, we explain how the maximum cross-correlation and time-frequency distance are computed. We note that this information is not available in the original paper and was obtained via private communication with the authors.

(0) As input we have two aligned audio snippets x and y of equal length L with a sampling rate r.

(1) x and y are normalized as (superscript T denotes transpose):

$$x' = \frac{x}{\sqrt{x^T \cdot x}} \qquad y' = \frac{y}{\sqrt{y^T \cdot y}} \tag{6}$$

Here, the denominator represents a square root of the signal's energy.

(2) The *maximum cross-correlation* between the normalized audio snippets x' and y' is computed as (we omit prime superscripts in $\hat{C}_{x,y}(l)$ for simplicity):

$$\hat{C}_{x,y}(l) = \max(|C_{x,y}(l)|) = \max\left(\left|\sum_{i=0}^{N-1} x'(i) \cdot y'(i-l)\right|\right) \tag{7}$$

$|\cdot|$ denotes the absolute value, N is the number of samples in audio snippets, and the lag l is set to the default value $2N - 1$ [35]. The resulting maximum cross-correlation is bounded within a range $\hat{C}_{x,y}(l) \in [0, 1]$, because we take the absolute value $|C_{x,y}(l)|$.

(3) To compute the frequency distance between audio snippets x and y a fast Fourier transform (FFT) weighted by a Hamming window is applied:

$$X = FFT(HW(x)) \qquad Y = FFT(HW(y)) \tag{8}$$

(4) Since the FFT is symmetric, only a half of the FFT values is taken to construct frequency vectors for x and y:

$$X_h = \left|X[1, \frac{L_X}{2}]\right| \qquad Y_h = \left|Y[1, \frac{L_Y}{2}]\right| \tag{9}$$

$|\cdot|$ denotes the absolute value, L_X and L_Y are lengths of FFT vectors X and Y.

(5) Frequency vectors X_h and Y_h are normalized similarly to step (1):

$$X'_h = \frac{X_h}{\sqrt{X_h^T \cdot X_h}} \qquad Y'_h = \frac{Y_h}{\sqrt{Y_h^T \cdot Y_h}} \tag{10}$$

(6) The frequency distance between audio snippets x and y is given by:

$$D_{f,xy} = \sqrt{\sum \left((X'_h - Y'_h) * (X'_h - Y'_h)\right)} \tag{11}$$

$*$ denotes element-wise multiplication.

(7) The time distance between audio snippets x and y is given by:

$$D_{t,xy} = 1 - \hat{C}_{x,y}(l) \tag{12}$$

(8) The *time-frequency distance* between audio snippets x and y is given by:

$$D_{tf,xy} = \sqrt{D_{t,xy}^2 + D_{f,xy}^2} \tag{13}$$

A.4.3 Machine Learning. After calculating these features over their dataset, Truong *et al.* used the machine learning suite *Weka* [12] using Multiboost [38] with grafted C4.5 decision trees [37] as weak learners in their evaluation. As Weka does not support large datasets, we chose to use the H2O framework [33] instead. For the training of the classifiers, we set the seed to 1619 and the early stopping to 5 rounds. This means that the training is repeatable when using the same seed and dataset, and the system will consider learning complete once no improvements have been made for five iterations. We let H2O train a set of independent models and perform a hyperparameter search to optimize the parameters (e.g., number of trees in the random forest) for the dataset, maximizing the cross-validated AUC. Afterwards, we select the top performing model and determine its EER as described in Section 4.4.

A.5 Shrestha *et al.*

The scheme by Shrestha *et al.* [28] utilizes ambient temperature, humidity, pressure, and precision gas collected by two devices to compute a number of context features, which are then fed into a machine learning classifier that outputs a prediction if these devices are colocated. Similarly to Truong *et al.*, this scheme addresses relay attacks by providing colocation evidence between two devices involved in ZIA. In this work, we focus on computing context features and obtaining classification results from the machine learning algorithms and we do not target a specific use case of thwarting relay attacks.

We first provide notations adopted from the original paper in Table 18. Second, we describe how different context features are computed. Third, we provide details of our machine learning methodology, where we discuss our datasets, the parameters of machine learning algorithms that we use and the evaluation procedure.

Due to a lack of hardware support, we were unable to collect precision gas and thus omit this context feature.

Table 18. Notations used by Shrestha *et al.*

Notation	Explanation
$s_a^{(k)}$	sample of context information k by device a
$D_{a,b}^{(k)}$	distance between samples of devices a and b

A.5.1 Context features. The authors convert ambient pressure P in millibars to altitude in meters using the following formula before computing context features.

$$h_{altitude} = \left(1 - \left(\frac{P_{station}}{1013.25}\right)^{0.190284}\right) * 145366.45 * 0.3048 \tag{1}$$

For each of the considered context information (ambient temperature, humidity, and altitude), the context feature is given by the absolute difference between two samples of context information collected devices a and b at time t:

$$D_{a,b}^{(k)} = |s_a^{(k)} - s_b^{(k)}| \tag{2}$$

A.5.2 Machine learning. The resulting distances are passed to a Multiboost classifier [38], with random forests [3] as a weak learner, using Weka [12]. The process for machine learning is identical to that described in the previous section.

B STUDY DESIGN

Table 19 presents hardware used to collect context information in car, office and mob/het scenarios.

Table 19. Sensing device used for data collection

Sensor type	Sensing device (sampling rate)			
	TI SensorTag CC2650 + Raspberry Pi 3 + Samson Go	Samsung Galaxy S6	Samsung Gear S3	RuuviTag+
Audio	16 kHz	16 kHz	16 kHz	–
Barometric pressure	10 Hz	5 Hz	10 Hz	10 Hz
Humidity	10 Hz	–	–	10 Hz
Luminosity	10 Hz	5 Hz	10 Hz	–
Temperature	10 Hz	–	–	10 Hz
BLE beacons	0.1 Hz	0.1 Hz	0.1 Hz	–
WiFi beacons	0.1 Hz	0.1 Hz	0.1 Hz	–
Accelerometer	10 Hz	50 Hz	50 Hz	–
Gyroscope	10 Hz	50 Hz	50 Hz	–
Magnetometer	10 Hz	50 Hz	50 Hz	–

– = sensor not available

Figure 20 contains a description of the device deployment in the car scenario, while Table 21 contains the mapping for the office scenario, and Table 22 shows device locations in the mob/het scenario.

Figure 10 shows the route the cars took during the car scenario (cf. Section 3.2). The route covers city traffic, country roads and highways between the cities of Darmstadt and Frankfurt in the state of Hesse in Germany (the actual GPS traces can be found in [9]).

Table 20. Device location mapping in the car scenario

Car 1 Device	Device location	Car 2 Device
01	Dashboard	07
02	Glove compartment	08
03	Between front seats	09
04	Right back handhold	10
05	Left back handhold	11
06	Trunk	12

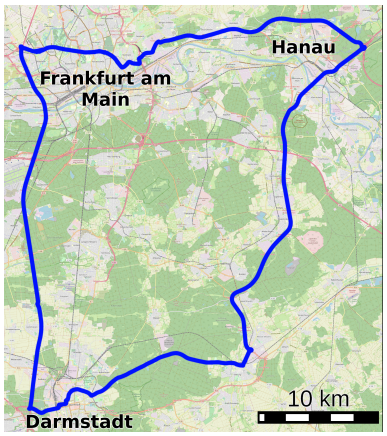


Fig. 10. Route driven in the car scenario

Table 21. Device location mapping in the office scenario

Office 1		Office 2		Office 3	
Dev.	Location	Dev.	Location	Dev.	Location
01	Near WiFi access point (*h*)	09	Screen of User 2 (*m*)	17	Wall behind Users 2 and 3 (*h*)
02	Window sill (*m*)	10	Window sill (*m*)	18	Window sill (*m*)
03	Above door to Office 2 (*h*)	11	Above door to Office 1 (*h*)	19	Lamp above User 1 (*h*)
04	Lamp above User 1 (*h*)	12	Lamp above User 1 (*h*)	20	Screen of User 1 (*m*)
05	Screen of User 1 (*m*)	13	Right screen of User 1 (*m*)	21	Screen of User 2 (*m*)
06	Screen of User 2 (*m*)	14	Left screen of User 1 (*m*)	22	Screen of User 3 (*m*)
07	In the cupboard (*h*)	15	In the cupboard (*l*)	23	Shelf next to the door (*m*)
08	Wall next to the door (*h*)	16	Shelf left of the door (*m*)	24	In the cupboard (*h*)

h = high position; *m* = medium position; *l* = low position

Table 22. Device location mapping in the mob/het scenario, initial configuration

Office 1		Office 2		Office 3	
Dev.	Location	Dev.	Location	Dev.	Location
01	Screen of User 1 (*m*)	11	Screen left from User 3 (*m*)	18	Screen in front of User 4 (*m*)
02	Screen of User 2 (*m*)	12	Screen of User 3 (*m*)	19	Screen of User 4 (*m*)
03	Near a power plug (*l*)	13	Near a power plug (*l*)	20	Near a power plug (*l*)
04	On top of robot station (*l*)	14	Near a fan (*l*)	21	On top of coffee machine (*m*)
05	Smartphone of User 1†	15	Smartphone of User 3†	22	Smartphone of User 4†
06	Smartwatch of User 1†	16	Smartwatch of User 3†	23	Smartwatch of User 4†
07	Laptop of User 1†	17	Laptop of User 3†	24	Laptop of User 4†
08	Smartphone of User 2†				
09	Smartwatch of User 2†				
10	Laptop of User 2†				
25	Smartphone on top of robot†				

h = high position; *m* = medium position; *l* = low position; † = mobile device

REFERENCES

[1] ANSI/ASA S1.11 2004. *Specification for Octave-Band and Fractional-Octave-Band Analog and Digital Filters.* Standard. American National Standards Institute.

[2] Fabien C. Y. Benureau and Nicolas P. Rougier. 2018. Re-run, Repeat, Reproduce, Reuse, Replicate: Transforming Code into Scientific Contributions. *Frontiers in Neuroinformatics* 11 (2018), 69.

[3] Leo Breiman. 2001. Random Forests. *Machine Learning* 45, 1 (2001), 5–32.

[4] Arne Brüsch, Ngu Nguyen, Dominik Schürmann, Stephan Sigg, and Lars Wolf. 2018. On the Secrecy of Publicly Observable Biometric Features: Security Properties of Gait for Mobile Device Pairing. *CoRR* abs/1804.03997 (2018).

[5] Mahmoud Elkhodr, Seyed Shahrestani, and Hon Cheung. 2016. The Internet of Things: New Interoperability, Management and Security Challenges. *International Journal of Network Security and its Applications* 8, 2 (2016), 85–102.

[6] Manuel Fernández-Delgado, Eva Cernadas, Senén Barro, and Dinani Amorim. 2014. Do We Need Hundreds of Classifiers to Solve Real World Classification Problems. *J. Mach. Learn. Res* 15, 1 (2014), 3133–3181.

[7] Mikhail Fomichev, Flor Álvarez, Daniel Steinmetzer, Paul Gardner-Stephen, and Matthias Hollick. 2018. Survey and Systematization of Secure Device Pairing. *IEEE Communications Surveys Tutorials* 20, 1 (2018), 517–550.
[8] Mikhail Fomichev, Max Maass, Lars Almon, Alejandro Molina, and Matthias Hollick. 2019. Audio Data from Mobile Scenario from "Perils of Zero-Interaction Security in the Internet of Things". https://doi.org/10.5281/zenodo.2537984
[9] Mikhail Fomichev, Max Maass, Lars Almon, Alejandro Molina, and Matthias Hollick. 2019. Index of Supplementary Files from "Perils of Zero-Interaction Security in the Internet of Things". https://doi.org/10.5281/zenodo.2537721
[10] Jerome H Friedman. 2001. Greedy Function Approximation: a Gradient Boosting Machine. *Annals of statistics* (2001), 1189–1232.
[11] Futurae Technologies AG. 2017. Futurae Authentication Suite. https://www.futurae.com/product/strongauth/ [Online, Accessed 2018-04-25].
[12] Mark A Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. 2009. The WEKA Data Mining Software: an Update. *SIGKDD Explorations* 11, 1 (2009), 10–18.
[13] Jun Han, Albert Jin Chung, Manal Kumar Sinha, Madhumitha Harishankar, Shijia Pan, Hae Young Noh, Pei Zhang, and Patrick Tague. 2018. Do You Feel What I Hear? Enabling Autonomous IoT Device Pairing Using Different Sensor Types. In *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 836–852.
[14] Nikolaos Karapanos, Claudio Marforio, Claudio Soriente, and Srdjan Capkun. 2015. Sound-Proof: Usable Two-Factor Authentication Based on Ambient Sound. In *USENIX Security Symposium*. 483–498.
[15] Chucri A. Kardous and Peter B. Shaw. 2014. Evaluation of Smartphone Sound Measurement Applications. *The Journal of the Acoustical Society of America* 135, 4 (apr 2014), EL186–EL192.
[16] C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas. 2017. DDoS in the IoT: Mirai and Other Botnets. *Computer* 50, 7 (2017), 80–84. https://doi.org/10.1109/MC.2017.201
[17] Hong Lu, Wei Pan, Nicholas D. Lane, Tanzeem Choudhury, and Andrew T. Campbell. 2009. SoundSense: Scalable Sound Sensing for People-Centric Applications on Mobile Phones. In *Proceedings of the 7th international conference on Mobile systems, applications, and services - Mobisys '09*. ACM Press, New York, New York, USA, 165.
[18] Nicolas Maisonneuve, Matthias Stevens, Maria E. Niessen, and Luc Steels. 2009. NoiseTube: Measuring and Mapping Noise Pollution with Mobile Phones. In *Information technologies in environmental engineering*. Springer, 215–228.
[19] Shrirang Mare, Andrés Molina Markham, Cory Cornelius, Ronald Peterson, and David Kotz. 2014. Zebra: Zero-effort Bilateral Recurring Authentication. In *Security and Privacy (SP), 2014 IEEE Symposium on*. IEEE, 705–720.
[20] Markus Miettinen, N Asokan, Thien Duc Nguyen, Ahmad-Reza Sadeghi, and Majid Sobhani. 2014. Context-based Zero-Interaction Pairing and Key Evolution for Advanced Personal Devices. In *ACM Conference on Computer and Communications Security (CCS)*. ACM, 880–891.
[21] Emiliano Miluzzo, Michela Papandrea, Nicholas D Lane, Hong Lu, and Andrew T Campbell. 2010. Pocket, Bag, Hand, etc. - Automatically Detecting Phone Context through Discovery. *PhoneSense 2010: International Workshop on Sensing for App Phones (November 2, 2010), held at ACM SenSys '10 (Zurich, Switzerland, November 2-5, 2010)* (2010), 21–25.
[22] Charith Perera, Arkady Zaslavsky, Peter Christen, and Dimitrios Georgakopoulos. 2014. Context Aware Computing for the Internet of Things: A Survey. *IEEE communications surveys & tutorials* 16, 1 (2014), 414–454.
[23] Dominik Schürmann, Arne Brüsch, Stephan Sigg, and Lars Wolf. 2017. BANDANA - Body Area Network Device-to-device Authentication Using Natural gAit. In *IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, 190–196.
[24] Dominik Schürmann and Stephan Sigg. 2013. Secure Communication Based on Ambient Audio. *IEEE Transactions on mobile computing* 12 (2013), 358–370.
[25] Carlton Shepherd, Iakovos Gurulian, Eibe Frank, Konstantinos Markantonakis, Raja Naeem Akram, Emmanouil Panaousis, and Keith Mayes. 2017. The Applicability of Ambient Sensors as Proximity Evidence for NFC Transactions. In *2017 IEEE Security and Privacy Workshops (SPW)*. IEEE, 179–188.
[26] Babins Shrestha, Manar Mohamed, and Nitesh Saxena. 2016. Walk-Unlock: Zero-Interaction Authentication Protected with Multi-Modal Gait Biometrics. *CoRR* abs/1605.00766 (2016).
[27] Babins Shrestha, Manar Mohamed, Sandeep Tamrakar, and Nitesh Saxena. 2016. Theft-Resilient Mobile Wallets: Transparently Authenticating NFC Users with Tapping Gesture Biometrics. In *Proceedings of the 32nd Annual Conference on Computer Security Applications*. ACM, 265–276.
[28] Babins Shrestha, Nitesh Saxena, Hien Thi Thu Truong, and N Asokan. 2014. Drone to the Rescue: Relay-resilient Authentication Using Ambient Multi-Sensing. In *International Conference on Financial Cryptography and Data Security (FC)*. Springer, 349–364.
[29] Babins Shrestha, Nitesh Saxena, Hien Thi Thu Truong, and N Asokan. 2018. Sensor-based Proximity Detection in the Face of Active Adversaries. *IEEE Transactions on Mobile Computing* (2018).
[30] Babins Shrestha, Maliheh Shirvanian, Prakash Shrestha, and Nitesh Saxena. 2016. The Sounds of the Phones: Dangers of Zero-Effort Second Factor Login based on Ambient Audio. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 908–919.

[31] Stephan Sigg. 2011. Context-based Security: State of the Art, Open Research Topics and a Case Study. In *Proceedings of the 5th ACM International Workshop on Context-Awareness for Self-Managing Systems*. ACM, 17–23.
[32] Fei Tang and Hemant Ishwaran. 2017. Random Forest Missing Data Algorithms. *Statistical Analysis and Data Mining: The ASA Data Science Journal* 10, 6 (2017), 363–377.
[33] The H2O.ai team. 2015. H2O: Scalable Machine Learning. http://www.h2o.ai version 3.1.0.99999 [Online, Accessed 2018-04-25].
[34] The MathWorks, Inc. 2018. Bandpass IIR Filter. https://mathworks.com/help/signal/ref/designfilt.html [Online, Accessed 2018-04-25].
[35] The MathWorks, Inc. 2018. Cross-correlation. https://mathworks.com/help/signal/ref/xcorr.html#bual1fd-maxlag [Online, Accessed 2018-04-25].
[36] Hien Thi Thu Truong, Xiang Gao, Babins Shrestha, Nitesh Saxena, N Asokan, and Petteri Nurmi. 2014. Comparing and Fusing Different Sensor Modalities for Relay Attack Resistance in Zero-Interaction Authentication. In *IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, 163–171.
[37] Geoffrey I. Webb. 1999. Decision Tree Grafting from the All-tests-but-one Partition. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence - Volume 2 (IJCAI'99)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 702–707.
[38] Geoffrey I. Webb. 2000. MultiBoosting: A Technique for Combining Boosting and Wagging. *Machine Learning* 40, 2 (2000), 159–196.
[39] Wei Xi, Chen Qian, Jinsong Han, Kun Zhao, Sheng Zhong, Xiang-Yang Li, and Jizhong Zhao. 2016. Instant and Robust Authentication and Key Agreement among Mobile Devices. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 616–627.

Received August 2018; revised November 2018; accepted January 2019