IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 2, 2019, accepted December 29, 2019, date of publication January 6, 2020, date of current version January 15, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2964151

Secure Accelerometer-Based Pairing of Mobile Devices in Multi-Modal Transport

BOGDAN GROZA[1], ADRIANA BERDICH[1], CAMIL JICHICI[1], AND RENE MAYRHOFER[2]
[1]Faculty of Automatics and Computers, Politehnica University of Timisoara, 300223 Timisoara, Romania
[2]Institute of Networks and Security and LIT Secure and Correct Systems Lab, Johannes Kepler University Linz, 4040 Linz, Austria

Corresponding author: Bogdan Groza (bogdan.groza@aut.upt.ro)

This work was supported by the Ministry of Research and Innovation, CNCS-UEFISCDI, under Project number PN-III-P1-1.1-TE-2016-1317, within PNCDI III (2018 to 2020).

ABSTRACT We address the secure pairing of mobile devices based on accelerometer data under various transportation environments, e.g., train, tram, car, bike, walking, etc. As users commonly commute by several transportation modes, extracting session keys from various scenarios to secure the private network of user's devices or even the public network formed by devices belonging to distinct users that share the same location is crucial. The main goal of our work is to establish the amount of entropy that can be collected from these environments in order to determine concrete security bounds for each environment. We test several signal processing techniques on the extracted data, e.g., low-pass and high-pass filters, then apply sigma-delta modulation in order to expand the size of the feature vectors and increase both the pairing success rate and security level. Further, we bootstrap secure session keys by the use of existing cryptographic building blocks EKE (Encrypted Key Exchange) and SPEKE (Simple Password Exponential Key Exchange). We implement our proof-of-concept application on Android smart-phones and take benefit from numerical processing environments for the off-line analysis of the collected datasets.

INDEX TERMS Authentication, cryptography, microcontrollers, network security.

I. INTRODUCTION AND MOTIVATION

As smart phones and other small, mobile, or embedded ''IoT'' are already ubiquitous, creating spontaneous connections between devices that share the same environment is an immediate necessity to facilitate local interaction. Harvesting environmental data to extract secure session keys that are shared between devices is relevant as it saves time and avoids the security inconveniences of poorly chosen passwords. While there are many research works that focus on harvesting environmental data to create such connections, there are still no practical deployments at a large scale suggesting that more research in this direction is welcome.

Our work addresses several transportation environments, e.g., by humans, bicycles as well as by motorized vehicles, etc. Each setting provides distinct patterns for the accelerometer data and measuring the amount of entropy that can be collected is necessary in order to confirm that a secure session key can be extracted. Specifically, we collect data from the following common types of transportation environments: i) trains, the heavier form of rail passenger transportation, ii) tram, the lighter form of rail transportation, mainly used inside cities, iii) cars, a transportation mode that offers a high degree of autonomy inside or outside cities, iv) bike, an increasingly widely used form of transportation inside cities, relying mostly on human power, and v) walking since it represents one of the most practical scenarios. We also add device shaking as a baseline scenario, as there are numerous works that address device pairing based on shaking patterns and a comparison between such a static and transport scenarios highlight the impact on entropy extraction. Moreover, even involuntarily, users may shake the devices in any of the previous transportation environments.

Figure 1 gives an abstract depiction of the addressed scenario. A user carries their devices (a smartphone, a smartwatch or a tablet, etc.) and these form a *private network* of the user. The user's private devices may pair based on accelerometer data from environments that are more intrinsic to the user (such as walking or riding a bike) and they may share an extensive history of accelerometer data. The commonly shared history can make bootstrapping a session key even more secure by the use cryptographic ratcheting or similar

The associate editor coordinating the review of this manuscript and approving it for publication was Kashif Saleem.

TABLE 1. Devices from our experiments and specifications for accelerometer sensors.

Device	Android	CPU	Memory	Acceleration Sensor
LG Optimus P700	4.0.3	1.0 GHz Cortex-A5	4 GB (2.4 GB user available) 512 MB RAM	BMA250 (Bosch Sensortec), Maximum Range - 39.22, Minimum Delay - 10 μs, Power - 0.03, Resolution - 0.038300782
Samsung J5	5.1.1	Quad-core 1.2 GHz Cortex-A53	8 GB, 1.5 GB RAM	K2HH (STM), Maximum Range - 39.2266, Minimum Delay - 10 μs, Power - 0.13, Resolution - 0.038307227
Samsung Galaxy A3	5.0.2	Quad-core 1.2 GHz Cortex-A53	16 GB, 1.5 GB RAM	BMC150 (Bosch Sensortec), Maximum Range - 19.6133, Minimum Delay - 10 μs, Power - 0.13, Resolution - 0.009576807

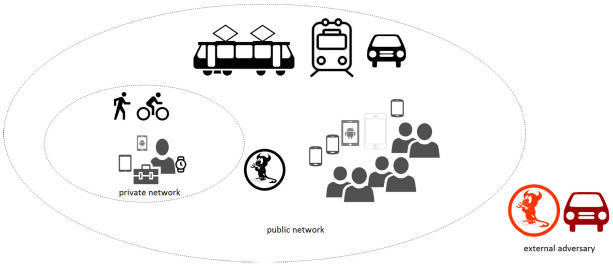

FIGURE 1. Overview of the addressed scenario with adversaries inside and outside the environment.

key continuity methods. On the other hand, the user may commute by train, tram or car and meet other users with which they may form a *public network* for exchanging various data such as contact information, photos, etc. Acceleration patterns from these environments can be used to bootstrap session keys for the public network. An adversary may be present both as a user in the same environment trying to tamper with the personal network or outside the environment trying to tamper with the public network, both these situations are depicted in Figure 1 and more details on the adversary capabilities are given in a forthcoming section. To answer to this adversarial model, besides a theoretical approximation of the adversary success rate, we add experiments in which mobile phones are placed in distinct locations inside a car or train, and we even add a car following scenario in which we collect data from a car that closely follows another. Note that this adversarial model goes beyond a honest-but-curious user and implicitly includes more targeted attacks by informed adversaries. However, on-device adversaries with perfect access to the accelerometer readings of a target user's device are considered out of scope of our current analysis.

We consider such an investigation, that covers multi-modal transport, to be necessary since today individuals usually commute between several transportation modes to reach a particular destination. Nonetheless, they may carry more than a single mobile device with them or may want to pair their device with another one from the same transportation environment.

Figure 2 depicts some of the transportation modes from which we collected data in our experiments and two of the mobile phones that we used, an LG Optimus L7 P700 and a Samsung J5, placed on the glove compartment of a car. A third phone, a Samsung A3, was also used in some of

FIGURE 2. Some transportation modes from our experiments (i-iv) and two of the phones (LG Optimus and Samsung J5) inside the glove compartment (v).

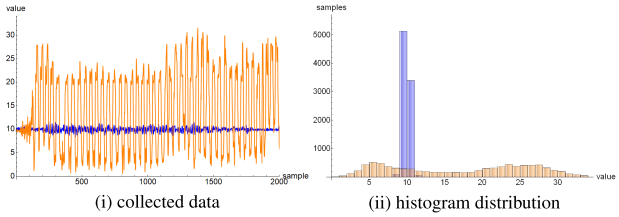

FIGURE 3. Data collected inside a tram (blue) vs. data collected during device shaking (orange) and histogram distribution (with LG Optimus phone).

the scenarios. Table 1 summarizes the specifications for the three phones, highlighting differences in computational resources and accelerometer specifications.

There are several technical challenges in pairing devices based on accelerometer data and multi-modal transport seems to complicate the problem even further. We now briefly discuss some concerns. First, as expected, different transportation environments lead to distinct acceleration patterns. In Figure 3 we contrast data recorded inside a tram with data recorded during shaking of the device by a person. Not surprisingly, accelerations inside the tram cover a narrow interval so an immediate question is whether there is enough entropy to extract a secure session key. Device shaking has been previously proposed to extract or validate session keys, (e.g. [3], [12], [21], [25]), and it is assumed that it provides a

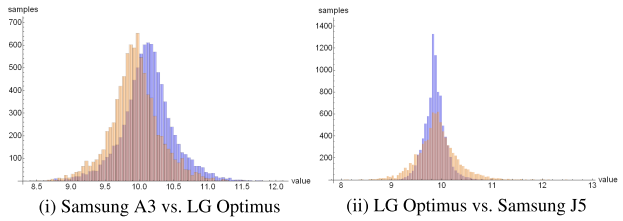

FIGURE 4. Histogram distribution of data collected from Samsung A3, LG Optimus and Samsung J5 in the same transportation environment (tram).

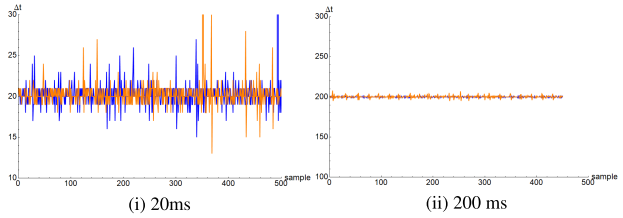

FIGURE 5. Drifts in the sampling time on LG Optimus at 20ms and 200ms in the same transportation environment (tram).

rich source of entropy, but some transportation environments may provide much weaker sources of entropy. Second, it is well known that due to physical imperfections of the sensors, data collected from distinct accelerometers is not identical. In fact, previous research works have shown that accelerometer sensor are so unique that it is possible to fingerprint devices based on data extracted from accelerometers [8]. Figure 4 shows the histogram distribution of data collected from an LG Optimus compared to a Samsung A3 in the same environment (left), then it compares the histogram of data from the LG Optimus with a Samsung J5 in the same environment (right). The Samsung A3 and LG Optimus are equipped with sensors from BOSCH, BMC150 and BMA250, while the J5 has a K2HH sensor from STM. For the sensors from the same manufacturer (on A3 and Optimus) there is only a slight bias between the distributions (left side of the figure). The J5 however has a less sensitive sensor and the histogram is much narrower, the recorded values do not cover the same interval as in case of the Optimus and A3 (left side of the figure). A third factor is that not all devices can cope with fast acquisition rates. In case of the LG Optimus, increasing the sampling rate to 50Hz (which is supported by the sensor according to the specs) led to numerous drifts in the timestamps of the collected data. Figure 5 shows drifts at 50Hz (20ms per sample) compared to drifts at 5Hz (200ms per sample). In the first case, drifts of 50%-100% are common and these have a high impact since the phones do not sample data at the same time (i.e., at 50Hz for each drift of 20ms one phone will remain one sample behind). For a sampling rate of 5Hz (right side of the figure) the sample time is quite stable. Fortunately, with the exception of the shaking scenario, most of the transportation environments provided slower changes in the shaking patterns and our experiments showed that we can rely even on a slow 5Hz sampling rate for most of the cases.

To respond to this heterogeneity caused by environments, sensors and nonetheless oscillators or other device characteristics, we test several techniques for extracting a common session key. In particular we use high-pass and low-pass (smoothness) filters, then we apply sigma-delta modulation over the data for extracting feature vectors. Finally, we rely on a secure key-exchange protocol that is guessing resilient, i.e., the Encrypted Key Exchange (EKE) [1], [2] and one if its derivatives [15], which are known to achieve provable security.

A. RELATED WORK

Within the scope of this article, we are primarily concerned with device-to-device (D2D) authentication and therefore focus our discussion of related work on this aspect, mostly ignoring user-to-device (U2D) authentication methods that use similar sensor analysis techniques including accelerometers. For the latter aspect, we refer to recent surveys of U2D authentication [28], [38]. D2D authentication in general has also been receiving increasing attention in the last decade and we refer to other surveys for a good overview of previously suggested approaches [6], [11], [19], [24]. Surveyed pairing methods include different sources of shared sensor data such as light, sound, etc.

Specifically for shaking-based acceleration pairing, [32] addresses pairing over Bluetooth for multiple wearable devices based on accelerometer data and the Martini Synch protocol from [17] uses accelerometer data for pairing two devices over Bluetooth and extracts 9-20 bits of entropy per second. However, it relies on fuzzy cryptography which is more difficult and costly to implement. More recently, the proposal from [33] similarly uses fuzzy cryptography to extract common keys from accelerometer data. Two protocols for pairing based on accelerometer data are discussed in [25]. One of the protocols is Diffie-Hellman based and the other is based solely in symmetric functions (the later version does not achieve guessing resilience). In [3] key generation is done based on the nearest-neighbor algorithm. Pairing based on accelerometer data for wearable devices is addressed in [37] and [16]. The work in [4] also add audio data (by using microphones and speakers) in addition to data collected from acceleration sensors.

Accelerometer-based pairing is also addressed in [36], a remote server is used for mediating the pairing operation (communication is done over TLS). The server checks if the datasets match and then the devices can connect with each other, which has the significant disadvantage of requiring online connectivity. In [42] machine-learning is used for generating secret shared keys between devices. A distinct methodology with heuristic trees and hash functions is proposed in [12], addressing the vulnerability of low-entropy vectors in man-in-the-middle attacks in a previous protocol proposal [23].

Our current work builds upon these previous ones by extending the commonly studied shaking-based scenario to different transport scenarios and focusing explicitly on

estimating the entropy that can be extracted in such modes. This is partially orthogonal to the concrete cryptographic pairing protocol used to establish a pairwise (or group) key. We therefore use a standard EKE-DH protocol here, which could potentially be replaced with computationally simpler (for low-end IoT style devices) or more involved (for addressing other threat model) variants.

In trusted environments, features from accelerometers can be used to determine if two devices are held by the same person [20], to detect driving patterns [35] or abnormal driving behaviour [22], [41] or to recognize human activities [30]. Distinguishing between different transportation modes based on accelerometer data, e.g., car, bike, bus, etc., has also gained momentum [10], [18], [29], [31], [40]. Recently, a large data-set for transportation mode recognition was made public by [39]. But in general, these works do not consider relevant adversary models. In terms of analyzing more advanced adversaries, robustness against active attacks in the shaking scenario (which we use as the baseline for security comparison in this work) has been demonstrated already [12], [21], [25]. However, sophisticated attacks to break the system, for instance, exploiting technical support by extracting acceleration automatically from video or also the analysis of the entropy of the shaking sequences is missing in most of these previous analysis, so that the picture of the security properties of shake-based pairing is still incomplete.

Other lines of work – not necessarily in the D2D authentication context – use accelerometer data in order to identify passenger seat location [14] or to monitor road conditions [5], [26]. Gaussian Mixture Models are used in [27] in order to authenticate persons to the smartphone based on the walking style.

II. PROCESSING ACCELEROMETER DATA AND SECURITY ANALYSIS

In this section, we describe the tools that we use for processing the data. Nonetheless, we make an assessment on the quality of the extracted data in terms of entropy and Hamming distances between feature vectors.

A. TOOLS FOR PROCESSING ACCELEROMETER DATA

In order to extract feature vectors (for the key-exchange protocol) from the accelerometer data collected on the phones, sensor data is passed through the alignment, scaling and filtering stages. The filtering stage consists in either applying a high-pass filter or a low-pass filter. The low-pass filter that we use is a moving average filter, also known as a smoothness filter. Additionally, we use sigma-delta modulation to expand accelerometer data in order to improve on the extracted entropy. We discuss technical aspects related to these in what follows. Accelerometer data in Android is provided in a Cartesian system, in all the discussions that follow we refer to the resulting acceleration from merging the results on the three axes as $a = \sqrt{a_x^2 + a_y^2 + a_z^2}$. Table 2 summarizes the notations that we use in the work.

TABLE 2. Summary of relevant notations.

$\bar{v}$	vector containing data from the accelerometer
$\bar{v}^0$	vector containing feature values of 0-Hamming distance from the accelerometer
b	one byte from the collected data
t	time-stamp for the accelerometer vector
Ham	Hamming distance
H	entropy
$\mathsf{H}_{\min}$	min entropy
p	success probability for matching one feature vector
$Adv\mathcal{D}$	adversary knowing the distribution of feature vectors
$Adv\mathcal{H}$	adversary knowing the distribution of 0-Hamming vectors
Usr	honest user
$\epsilon_{\blacklozenge}^{k,n}$	advantage for at least k out of n matches, where $\blacklozenge \in \{Adv\mathcal{D}, Adv\mathcal{H}, Usr\}$
p	large prime used for Diffie-Hellman Key exchange
g	a generator of Z_p

Temporal alignment. Before proceeding to the data collection stage, a loose time-synchronization between the devices is required in order to avoid deviations from the real-time clock. The loose time-synchronization that we use requires two simple steps between principals:

$$A \rightarrow B : t_{A,0}, \text{rnd}_{128}$$
$$B \rightarrow A : t_{B,0}, H(t_{A,0}, \text{rnd}_{128}, t_{B,0})$$

Here rnd_{128} denotes a random 128-bit value and $t_{A,0}$ is the recorded time on A's side when sending the first message while $t_{B,0}$ is the recorded time on B's side when sending the second message. For simplicity and since it is out of scope for the current work, the loose time-synchronization protocol has no security mechanisms. If an adversary tampers with it, accelerometer data will be misaligned and the best that the adversary could achieve is a DoS attack (this can be done anyway by injecting fake data). Still, if such an attack could cause concerns, security can be added to the time-synchronization protocol. Let $t_{A,1}$ be the time recorded on A's side when receiving the second message from B. Then $\xi = t_{A,1} - t_{A,0}$ is the maximum synchronization error and for any message received by A at time $t_{A,k}$, the clock on B's side will be $t_{B,k} \in [t_{B,0} + t_{A,k} - t_{A,0}, t_{B,0} + t_{A,k} - t_{A,0} + \xi]$. The synchronization error ξ should usually be in the order of milliseconds. Even after we performed this synchronization step, we determined that the sequences may have an offset of 1 sample and in rare cases of 2 samples. This can be easily detected and fixed since the devices can exchange the first sequence of collected data in plaintext just to remove this offset.

High-pass and smoothness filters. We did experiment with both high-pass and smoothness filters. The intuition behind using high-pass filters was that they are best for extracting the noise from the accelerometer data, thus increasing the chances for variations in the accelerometer data to occur in the key-exchange material. The intuition behind smoothness filters is the reverse, by applying such filters we remove unwanted noise from the data and the chances for correct pairing increase (as expected however, this has a slight negative impact on security). For smoothness, we used a moving average filter with a filtering window set to 10. Due to

Algorithm 1 Sigma-delta modulation

procedure SIGMA-DELTA MODULATION
  $\sigma \leftarrow 0.04$, $\delta \leftarrow 10$, $\epsilon \leftarrow 0.01$
  $\widetilde{b} \leftarrow \{\}$, $index \leftarrow 1$, $b \leftarrow 0$, $t \leftarrow 0$
  while $t < \max(\overline{\theta})$ **do**
    $i \leftarrow index$
    while $t > \overline{\theta}[i+1] \wedge i < |tst|$ **do**
      $i \leftarrow i + 1$
    end while
    if $|\overline{v}[i] - \widetilde{v}| < \epsilon$ **then** $b = 0$
    else
      if $\overline{v}[i] > \widetilde{v}$ **then** $b = 1$
      else $b = -1$
      end if
    end if
    $\widetilde{b} = \text{Append}(\widetilde{b}, b)$
    $\widetilde{v} = \widetilde{v} + b\sigma$
    $t = t + \delta$
  end while
end procedure

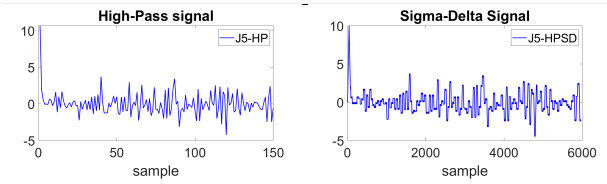

FIGURE 6. Algorithm for sigma-delta modulation (up), original signal from Samsung J5 and result (down).

space constraints, we skip details in these algorithms but we later discuss plots containing the results after filtering.

Sigma-delta modulation. Modulating the signal is essential in order to extract bits from the samples provided by the accelerometer. We opted for a ternary sigma-delta modulation where at each step we replace the signal with a 0 if the modulated signal is close enough to the original (by some error margin ϵ), or a ± 1 otherwise, i.e., in case that the modulated signal is smaller or bigger. The algorithm for sigma-delta modulation is depicted in Figure 6 and below it we show the original signal vs. the result. With sigma-delta modulation, by re-sampling at $\delta = 5ms$ over the original acquisition rate of $200ms$, we get 40 new samples for each original sample. Thus the 150 sampling points expand to $40 \times 150 = 6000$ samples on the second plot. Based on experiments we determined that $\delta = 10$ and $\epsilon = 0.01$ are good choices but these can be further modified according to the experimental data. We set σ to the standard deviation of the signal multiplied by 0.1. When sigma-delta modulation is not used, we simply center the signal over the mean and take the sign of each sample to extract a ± 1. This procedure indeed lowers the number of extracted bits as we discuss in the experiments, so sigma-delta is preferable.

Figure 7 shows plots with the original signal (left) and several processing techniques (right). The plots are for several of the experimental scenarios: train, tram, car, bike and shake. The scaled signal has a similar shape to the original one, a reason for which we omit plots for simple scaling of the signal. When using smoothness filters, the signal becomes more uniform while with high-pass filtering high frequency

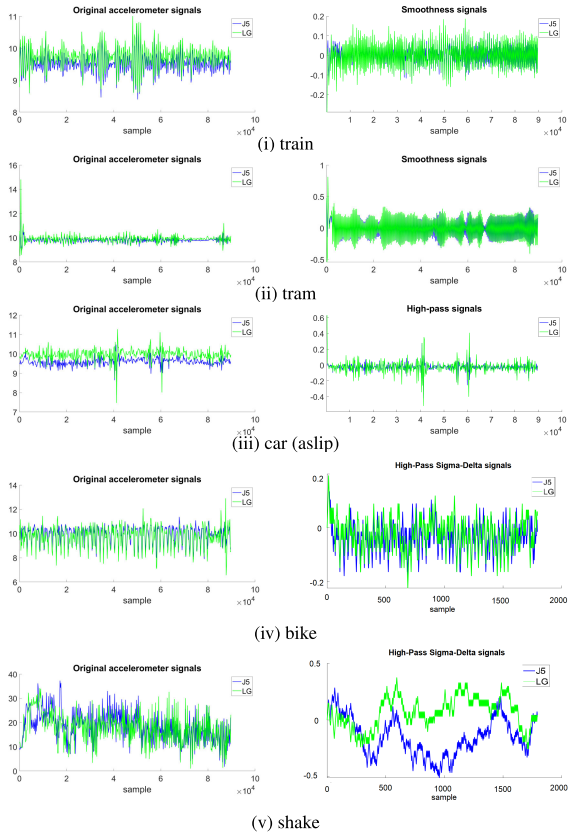

FIGURE 7. Accelerometer signals before (left) and after filtering (right) in various transportation environments.

noises amplify. The Sigma-Delta modulation gives a more discrete shape for the signal as can be seen in the last plots.

B. THEORETICAL FRAMEWORK FOR SECURITY ANALYSIS

Adversary model. In principle, guessing resilient protocols, such as EKE [2], preclude an adversary from mounting an off-line brute-force attack. However, the attacker may take an active role as a man-in-the-middle adversary and may inject fake data trying to claim that he is an honest party in the communication. Our adversary model does assume the existence of a regular Dolev-Yao [9] adversary that has full control of the communication channel and such an attack is realistic. Consequently, a concrete metric for the security level of the samples is needed. We consider that on-device adversaries that have access to high-quality accelerometer data stream – e.g. through malware being installed and running on the device in the background while a man-in-the-middle attack is concurrently performed – are out of scope for the current work since such an adversary can trivially extract the key similar to the genuine app. However, we note that an analysis of different types of on-device adversaries, e.g. considering Javascript from malicious websites sampling accelerometers at lower accuracy or sample rates, may be subject to future research and could enable protection even against lower-fidelity on-device adversaries.

Another type of adversary may be an attacker that stalks the honest user with the hope to harvest enough accelerometer

data to recover the secret key shared between his devices. In the experiments, we also add a car following scenario and show that the data does not support direct pairing of devices from distinct cars even if these are just meters away. Devices that share the same environment and exhibit the same accelerations are not directly viewed as adversaries but as part of the public peers. As stated in the introductory section these should separate from the devices carried by the same user based specific moves or a common history between the devices of the same user.

We expect that phones from the same environment will sense the same acceleration patterns. However, separation should occur when a user moves his gear inside the environment adding specific shaking patterns, e.g., the user may move inside the tram or train or move his backpack inside the car. We suggest that phones from the same environment may pair representing a sort of a *public* network of the user. In contrast, devices that are carried by the user pair under a *private* network of the user. While we are not aware of a consensus in current literature on such a distinction between adversaries, within the scope of this article we consider this explicit distinction between public networks (which might include adversaries) and private networks (in which we assume only trusted devices). If an adversary manages to attach one of his devices to the ones carried by the user, and therefore introduce it physically into the resulting private network, we assume those to successfully pair and thus consider such physical intrusions outside the scope of our analysis.

As a security metric, we choose to stay close to existing models from cryptography. Concretely, we use the *guessing probability* of a random variable X that represents one byte of information from accelerometers, i.e., $\gamma(X) = \max\{\Pr[X = b] : b \in [0, 2^8 - 1]\}$. From this measurement, the *min entropy* of variable X can be immediately computed as $\log_2(1/\gamma)$, see [34]. In what follows, we use these metrics for analyzing the security of accelerometer data.

Our pairing algorithm relies on feature vector from accelerometer data of Hamming distance equal to 0, i.e., when the same vector is recorded on both phones, a part of the key can be extracted. The feature vectors consist in either the sign of each sample (encoded as 0 or 1, for negative vs. positive) or in bits resulting from the sigma-delta modulation. The success probability for two honest parties to extract one such vector can be computed based on experimental data as:

$$p_{Usr} = \frac{|\overline{\mathsf{v}}^0|}{|\overline{\mathsf{v}}|},$$

where $\overline{\mathsf{v}} = \{b_1, b_2, \ldots, b_\ell\}$ denotes the vector of all collected bytes and $\overline{\mathsf{v}}^0 = \{b_1^0, b_2^0, \ldots, b_{\ell'}^0\}$ the vector of all bytes that have a Hamming-distance equal to zero. This leads to $p_{Usr} = \ell'/\ell$. We can also define the guessing probability of a byte b from $\overline{\mathsf{v}}^0$ as:

$$\gamma = \max\{\Pr[b = b_i] : i = 1..\ell', b_i \in \overline{\mathsf{v}}^0\}.$$

Based on this probability we can also define the minimum entropy of the feature vector with 0-Hamming distance $\overline{\mathsf{v}}^0$ as follows:

$$\mathsf{H}_{\mathsf{min}}^{\overline{\mathsf{v}}^0} = \sum_{i=1,\ell'} \log_2(\gamma) = \ell' \log_2(\gamma).$$

To these, we will also add the overall entropy of the feature vectors which is also commonly used in several of the related works (for a more intuitive treatment, in the following experiments we refine this to the number of extracted bits/second). This is defined as:

$$\mathsf{H}^{\overline{\mathsf{v}}^0} = -\sum_{i=1,\ell'} p_i \log_2(p_i), p_i = \Pr\left[b = b_i, b_i \in \overline{\mathsf{v}}^0\right].$$

To define the security level of the protocol, it is necessary to measure the adversary advantage (success probability) in claiming to be an honest user. To quantify this, we need an adversary model. The simplest assumption is that the adversary cannot infer anything except for the fact that each sample has 8 bits (this would trivially imply that guessing one value has probability 2^{-8}). However, this adversary model is weak especially if we consider that for calibration purposes the devices may exchange some samples in cleartext and thus the adversary has access at least to a limited dataset. On the other hand the adversary may be equipped with similar devices and be in possession of previous datasets from the same environment. It seems that one of the strongest assumptions that we can make about the adversary is that he has access to the set of samples that yield a 0 Hamming distance except that he does not know which of the samples will be selected. Let $Adv\mathcal{H}$ denote this adversary.

Concretely, considering the feature vectors as arrays $\overline{\mathsf{v}}^0 = \{b_1^0, b_2^0, \ldots, b_{\ell'}^0\}$ of ℓ' bytes, let $b_{\max}$ the byte occurring with the maximum probability in $\overline{\mathsf{v}}^0$ and $\ell_{\max}$ its number of occurrences in $\overline{\mathsf{v}}^0$ (in case there are multiple such values $\ell_{\max}$ is the same for all and what follows remains unchanged). Assuming that $Adv\mathcal{H}$ uses this byte to make his best guess, then the advantage of the adversary is:

$$p_{Adv\mathcal{H}} = \frac{\ell_{\max}}{\ell}.$$

This implies that the adversary has no knowledge of the current accelerometer value so the best he can do is to make the guess with the higher success rate based on what he knows about the bytes in $\overline{\mathsf{v}}^0$.

A weaker assumption on the adversary would be that it has no access to the vectors that yield a Hamming distance equal to zero, but that the adversary has access to the complete acceleration vectors $\overline{\mathsf{v}}$ and guesses that the value occurring with the highest probability in $\overline{\mathsf{v}}$ will be the one selected as having a zero Hamming distance. Let this adversary be denoted by $Adv\mathcal{D}$. Note however, that there may be multiple values that have the same higher probability in the feature

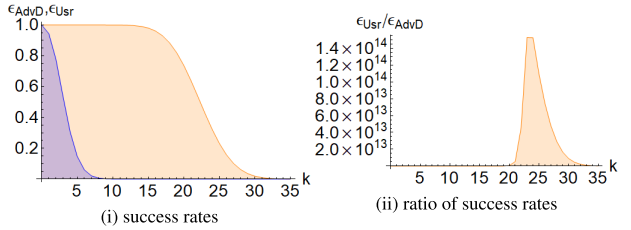

FIGURE 8. Honest users vs. adversaries based on data extracted on a train in a favourable scenario $p_{Usr} = 0.39$, $p_{Adv\mathcal{D}} = 0.05$: success probabilities $\epsilon_{Usr}^{56,\ell}$, $\epsilon_{Adv\mathcal{D}}^{56,\ell}$ (left) and ratio of success probabilities $\epsilon_{Usr}^{56,\ell}/\epsilon_{Adv\mathcal{D}}^{56,\ell}$ (right).

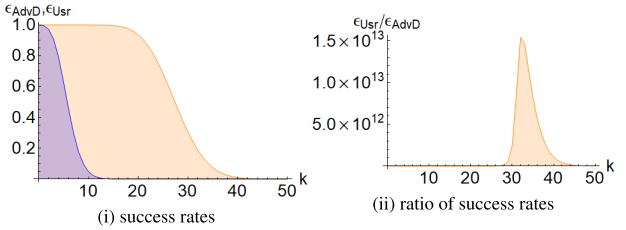

FIGURE 9. Honest users vs. adversaries based on data extracted during walking in a less favourable scenario $p_{Usr} = 0.05$, $p_{Adv\mathcal{D}} = 0.02$: success probabilities $\epsilon_{Usr}^{k,541}$, $\epsilon_{Adv\mathcal{D}}^{k,541}$ (left) and ratio of success probabilities $\epsilon_{Usr}^{k,541}/\epsilon_{Adv\mathcal{D}}^{k,541}$ (right).

vector $\bar{v}$ but it may be the case that these values do not have the same probability to occur in the set of features with Hamming distance equal to zero. If adversary $Adv\mathcal{D}$ selects the value that also appears in $\bar{v}^0$ with the highest probability, then $Adv\mathcal{D}$ is identical to $Adv\mathcal{H}$. To separate between the two types of adversaries, i.e., obtain an upper and lower bound for security, we assume that $Adv\mathcal{D}$ will make the poorer choice and go for the value that has the maximum probability in $\bar{v}$ but the lowest in $\bar{v}^0$. Let $\bar{v}_{\max}$ be the vector of byte values that have the maximum probability of occurrence in $\bar{v}$, then we can define the success of $Adv\mathcal{D}$ as:

$$p_{Adv\mathcal{D}} = \min\{\Pr[b = b_{\max}] : b \in \bar{v}^0, b_{\max} \in \bar{v}_{\max}\}.$$

To increase the security of the pairing process, we ask for matching at least k out-of ℓ trials (where ℓ is the total number of bytes extracted from the accelerometer). This leads to the following security bounds for honest users and adversaries:

$$\epsilon_{\blacklozenge}^{k,\ell} = \sum_{i=k,\ell} \binom{\ell}{i} p_{\blacklozenge}^{i}(1-p_{\blacklozenge})^{\ell-i} = 1 - \sum_{i=0,k-1} \binom{\ell}{i} p_{\blacklozenge}^{i}(1-p_{\blacklozenge})^{\ell-i}.$$

where $\blacklozenge \in \{Adv\mathcal{D}, Adv\mathcal{H}, Usr\}$. We try to clarify the applicability of these security bound by some practical examples that we encountered both in favourable and disadvantageous scenarios. Experimental data will be discussed in large in the next section, here we simply illustrate how the advantage of honest users and adversaries scales up. First, for mobile phones place along each other in a train we obtained a good success rate for honest users at $p_{Usr} = 0.39$ and a much lower success rate for our basic adversary at $p_{Adv\mathcal{D}} = 0.05$. Having a sampling rate of 200ms, during 90s we collect 56 vectors each of 8 bit (i.e., $8 \times 56 \approx 90\ s$). By setting $k = 9$ we get the honest user success rate at $\epsilon_{Usr}^{9,56} = 99.99\%$ while for the adversary the success rate is $\epsilon_{Adv\mathcal{H}}^{9,56} = 0.17\%$. Figure 8 shows the plot for the success rate over a wider choice of k as well as the ration between the success rate of the honest users and adversaries. In a less favourable scenario, data collected at 20ms intervals during walking, one of the phones (the LG Optimus) showed a lot of clock drifts during sampling. This may have been a contributing factor as we got a very small success probability $p_{Usr} = 0.05$ and $p_{Adv\mathcal{H}} = 0.01$. With proper tuning of the parameters, the previous bounds show that honest users still pair at higher chances than dishonest adversaries. Since sampling is done at 20ms, we have 541 samples in 90s, i.e., $20 \times 8 \times 541 \approx 90s$. By setting $k = 14$ we get $\epsilon_{Usr}^{14,541} = 99.82\%$ and $\epsilon_{Adv\mathcal{H}}^{14,541} = 0.13\%$. This turns out to be surprisingly close to our previous favourable scenario, although it will require exchanging more feature vectors. Again, Figure 9 shows the plot for the success rate over a wider choice of k as well as the ration between the success rate of the honest users and adversaries. So in principle, as long as the ratio of the success rate between honest users and adversaries remains sufficiently high, bootstrapping a secure session key is feasible.

C. ANALYZING EXPERIMENTAL DATA

A comprehensive discussion on experimental data now follows. First, in Figure 10 we show data collected in the seven use cases: train, tram, car (glove compartment and anti-slip), bike, walk and shake. The plots on the left side show the value of the first bytes recorded from the J5 accelerometers. The right side plot shows the value of the bytes from the set of bytes with Hamming-distance equal to 0 (note that there are fewer bytes than in the left side). The distribution of the bytes seems somewhat uniform which is good for security reasons. We further refine these results in Figure 11. On the left side, we show the histogram of Hamming distances. Notably, the train and tram have somewhat smaller Hamming distances while the worst result is from the car and the shaking process. On the right side of the figure, we show the guessing probability of the bytes scaled along with the Hamming distance. The probabilities are somewhat uniform which suggest that vectors with Hamming distance equal to 0 should be partially indistinguishable from the rest (which is desirable from security reasons).

In Tables 3 and 4 we summarize the results for all the experiments and all the filtering mechanisms. We give the results only from one of the phones since they were similar on the other. We separate between the simple filtering techniques and sigma-delta modulation since the later boosts the number of samples (due to time division by parameter δ).

For the filtering techniques in Table 3, it is easy to note that smoothness lowers the entropy to 2 bits and sometimes even to 0. This makes the guessing adversary win with probability 1 in some cases, so smoothness does not seem to

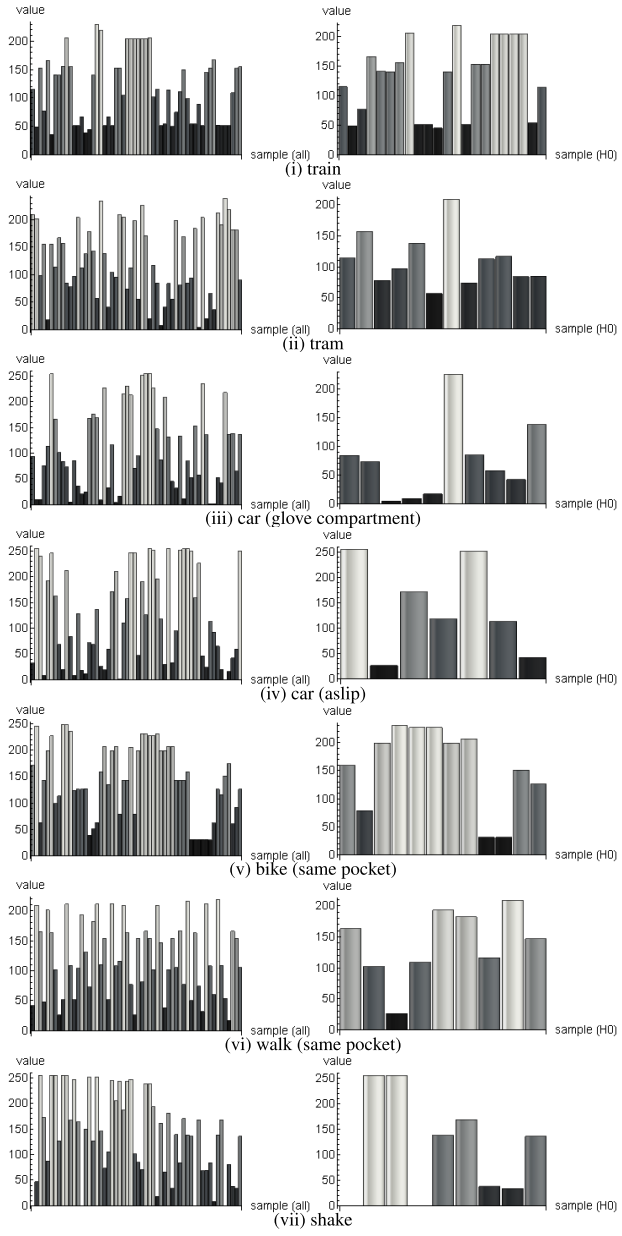

FIGURE 10. Accelerometer sample value as collected on J5 (left) and accelerometer values of matched samples for J5 and LG (right) for several uses cases.

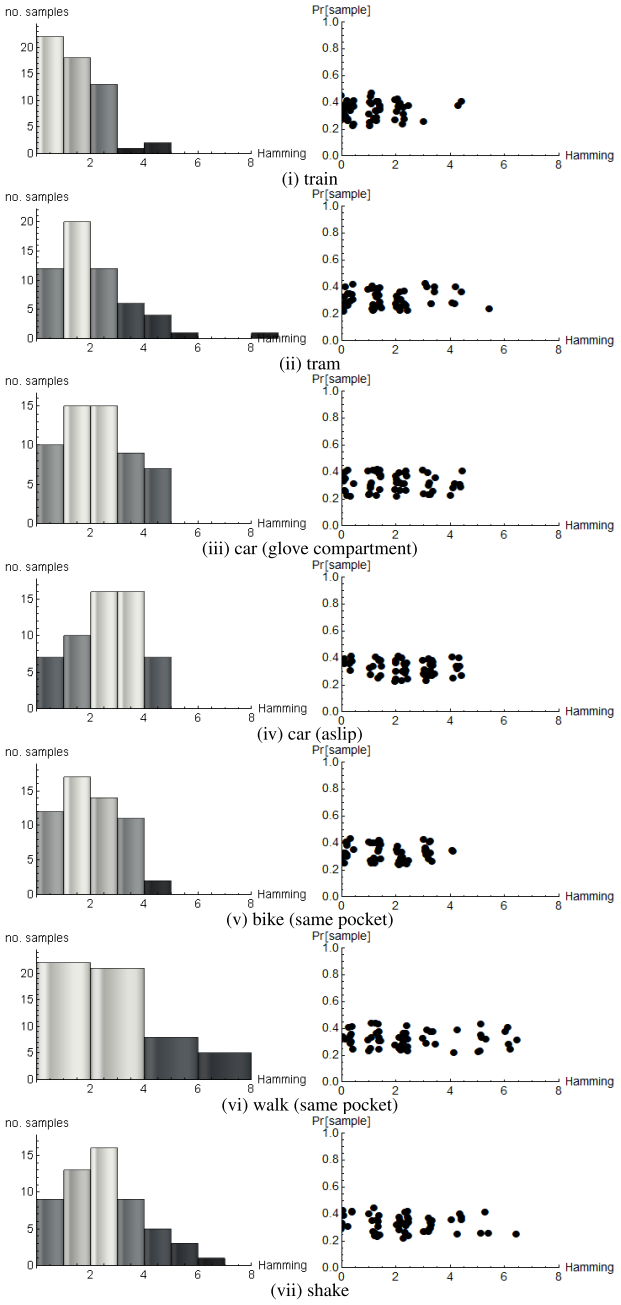

FIGURE 11. Histogram distribution of Hamming distances (left) and guessing probability (right) as recorded on Samsung J5 for several use cases.

help from a security perspective. Indeed, when smoothness is applied, the devices will be aways successfully paired, i.e., $p_{agen} = 1$. Simple scaling and high-pass filtering seem to provide results that are close to each other. With high-pass filtering, since it is more sensitive to changes in the signal, the pairing probability decreases but only by a small amount. In some cases high-pass results in a small increases the byte entropy but also entropy decreases in other scenarios (this is because we compute byte entropy w.r.t. to the vectors of Hamming-distance equal to 0 which apparently now have a smaller density). Generally however, the minimum entropy $\mathsf{H}_{\mathsf{min}}$ is 1-2 bits/byte and the success probability of honest parties remains high (82%–100% in case when the 7 out of 56 limit is imposed, i.e., $\ell/8$). The only exceptions are the car anti-slip scenario in case of simple scaling and the bike scenario in case of the high-pass filtering. Here the success probability for honest parties decreases to 0.56 and 0.39 respectively. The solution is to switch to the $\ell/16$ limit which requires at least 4 matches instead of 8. In this case, the probability of honest users increases to over 95% percents and that of the adversary remains only 7%.

We note that these values are extracted from 90 seconds of accelerometer data, in the long run (several minutes) specific parameters (k, n) can be chosen to reach negligible success rates for the adversary. We also display the entropy collected during each second in the

TABLE 3. Summary of experimental data with simple scaling (SS), high-pass filtering (HP) and smoothness (90 seconds for data collection on LG optimus).

Proc.	Env.	$\mathrm{P}_{Adv\mathcal{D}}$	$\mathrm{P}_{Adv\mathcal{H}}$	P_{Usr}	$\epsilon^{\ell/16}_{Adv\mathcal{D}}$	$\epsilon^{\ell/16}_{Adv\mathcal{H}}$	$\epsilon^{\ell/16}_{Usr}$	$\epsilon^{\ell/8}_{Adv\mathcal{D}}$	$\epsilon^{\ell/8}_{Adv\mathcal{H}}$	$\epsilon^{\ell/8}_{Usr}$	$\mathsf{H}_{\min}$	H	$\mu(\mathrm{Ham})$	$\lvert\bar{\mathbf{v}}_0\rvert$	ℓ
SS	Train	0.07	0.07	0.39	0.77	0.77	1.00	0.10	0.10	1.00	2.46	0.90	0.98	22	56
SS	Tram	0.02	0.02	0.21	0.08	0.08	1.00	$< 10^{-4}$	$< 10^{-4}$	0.97	3.58	0.48	1.63	12	56
SS	CarG	0.02	0.02	0.18	0.08	0.08	1.00	$< 10^{-4}$	$< 10^{-4}$	0.89	3.32	0.37	1.79	10	56
SS	CarA	0.00	0.02	0.13	0.00	0.08	0.98	0.00	$< 10^{-4}$	0.56	2.81	0.22	2.11	7	56
SS	Bike1	0.04	0.04	0.21	0.32	0.32	1.00	0.00	0.00	0.97	2.58	0.41	1.54	12	56
SS	Walk1	0.00	0.02	0.16	0.00	0.08	1.00	0.00	$< 10^{-4}$	0.82	3.17	0.32	2.30	9	56
SS	Shake	0.04	0.04	0.16	0.32	0.32	1.00	0.00	0.00	0.82	2.17	0.27	2.02	9	56
HP	Train	0.02	0.04	0.30	0.08	0.32	1.00	$< 10^{-4}$	0.00	1.00	3.09	0.71	1.05	17	56
HP	Tram	0.09	0.09	0.36	0.89	0.89	1.00	0.23	0.23	1.00	2.00	0.78	1.20	20	56
HP	CarG	0.02	0.02	0.21	0.08	0.08	1.00	$< 10^{-4}$	$< 10^{-4}$	0.97	3.58	0.48	1.36	12	56
HP	CarA	0.00	0.04	0.16	0.00	0.32	1.00	0.00	0.00	0.82	2.17	0.29	1.59	9	56
HP	Bike1	0.02	0.02	0.11	0.08	0.08	0.95	$< 10^{-4}$	$< 10^{-4}$	0.39	2.58	0.17	1.79	6	56
HP	Walk1	0.02	0.02	0.16	0.08	0.08	1.00	$< 10^{-4}$	$< 10^{-4}$	0.82	3.17	0.32	2.50	9	56
HP	Shake	0.09	0.09	0.21	0.89	0.89	1.00	0.23	0.23	0.97	1.26	0.35	2.41	12	56
SM	Train	0.77	0.77	0.84	1.00	1.00	1.00	1.00	1.00	1.00	0.13	0.31	0.18	47	56
SM	Tram	0.95	0.95	0.96	1.00	1.00	1.00	1.00	1.00	1.00	0.03	0.08	0.05	54	56
SM	CarG	0.09	0.11	0.39	0.89	0.95	1.00	0.23	0.39	1.00	1.87	0.65	1.25	22	56
SM	CarA	0.07	0.11	0.32	0.77	0.95	1.00	0.10	0.39	1.00	1.58	0.48	1.63	18	56
SM	Bike1	0.68	0.68	0.68	1.00	1.00	1.00	1.00	1.00	1.00	0.00	0.00	0.45	38	56
SM	Walk1	0.34	0.34	0.39	1.00	1.00	1.00	1.00	1.00	1.00	0.21	0.19	2.05	22	56
SM	Shake	0.18	0.18	0.41	1.00	1.00	1.00	0.89	0.89	1.00	1.20	0.51	1.32	23	56

TABLE 4. Summary of experimental data with sigma-delta modulation over simple scaling (SS), high-pass filtering (HP) and smoothness (90 seconds for data collection on Samsung J5).

Proc.	Env.	$\mathrm{P}_{Adv\mathcal{D}}$	$\mathrm{P}_{Adv\mathcal{H}}$	P_{Usr}	$\epsilon^{\ell/16}_{Adv\mathcal{D}}$	$\epsilon^{\ell/16}_{Adv\mathcal{H}}$	$\epsilon^{\ell/16}_{Usr}$	$\epsilon^{\ell/8}_{Adv\mathcal{D}}$	$\epsilon^{\ell/8}_{Adv\mathcal{H}}$	$\epsilon^{\ell/8}_{Usr}$	$\mathsf{H}_{\min}$	H	$\mu(\mathrm{Ham})$	$\lvert\bar{\mathbf{v}}_0\rvert$	ℓ
SSSD	Train	0.06	0.09	0.27	0.55	1.00	1.00	$< 10^{-26}$	$< 10^{-7}$	1.00	1.56	14.04	2.85	611	2240
SSSD	Tram	0.10	0.10	0.23	1.00	1.00	1.00	$< 10^{-5}$	$< 10^{-5}$	1.00	1.20	12.42	3.19	508	2240
SSSD	CarG	0.10	0.10	0.21	1.00	1.00	1.00	$< 10^{-6}$	$< 10^{-6}$	1.00	1.11	10.68	3.27	462	2240
SSSD	CarA	0.15	0.15	0.23	1.00	1.00	1.00	1.00	1.00	1.00	0.65	9.28	3.17	521	2240
SSSD	Bike1	0.06	0.06	0.26	0.48	0.65	1.00	$< 10^{-27}$	$< 10^{-25}$	1.00	2.00	15.38	2.42	576	2240
SSSD	Walk1	0.05	0.07	0.26	0.01	0.96	1.00	$< 10^{-40}$	$< 10^{-19}$	1.00	1.87	14.57	2.01	586	2240
SSSD	Shake	0.06	0.06	0.19	0.17	0.17	1.00	$< 10^{-32}$	$< 10^{-32}$	1.00	1.76	10.08	2.05	436	2240
HPSD	Train	0.11	0.11	0.23	1.00	1.00	1.00	0.03	0.03	1.00	1.02	11.39	3.43	507	2240
HPSD	Tram	0.07	0.08	0.23	0.94	1.00	1.00	$< 10^{-19}$	$< 10^{-14}$	1.00	1.55	10.25	3.42	510	2240
HPSD	CarG	0.11	0.11	0.24	1.00	1.00	1.00	0.00	0.00	1.00	1.18	11.30	3.47	536	2240
HPSD	CarA	0.10	0.10	0.24	1.00	1.00	1.00	0.00	0.00	1.00	1.22	11.05	3.46	548	2240
HPSD	Bike1	0.06	0.06	0.19	0.22	0.48	1.00	$< 10^{-31}$	$< 10^{-27}$	1.00	1.63	10.38	3.34	431	2240
HPSD	Walk1	0.05	0.07	0.24	$< 10^{-3}$	0.98	1.00	$< 10^{-48}$	$< 10^{-17}$	1.00	1.69	14.05	2.57	530	2240
HPSD	Shake	0.08	0.08	0.23	1.00	1.00	1.00	$< 10^{-13}$	$< 10^{-11}$	1.00	1.44	11.31	2.17	506	2240
SMSD	Train	0.22	0.22	0.26	1.00	1.00	1.00	1.00	1.00	1.00	0.27	6.68	2.93	581	2240
SMSD	Tram	0.03	0.05	0.16	$< 10^{-11}$	$< 10^{-4}$	1.00	$< 10^{-77}$	$< 10^{-49}$	1.00	1.80	10.01	3.06	358	2240
SMSD	CarG	0.46	0.46	0.48	1.00	1.00	1.00	1.00	1.00	1.00	0.08	4.55	1.85	1083	2240
SMSD	CarA	0.44	0.44	0.47	1.00	1.00	1.00	1.00	1.00	1.00	0.10	5.81	1.82	1060	2240
SMSD	Bike1	0.02	0.03	0.10	$< 10^{-40}$	$< 10^{-12}$	1.00	$< 10^{-151}$	$< 10^{-79}$	$< 10^{-4}$	1.60	6.19	3.15	222	2240
SMSD	Walk1	0.06	0.07	0.25	0.65	0.81	1.00	$< 10^{-25}$	$< 10^{-23}$	1.00	1.88	14.44	2.45	551	2240
SMSD	Shake	0.10	0.10	0.22	1.00	1.00	1.00	0.00	0.00	1.00	1.07	7.82	2.23	490	2240

H column. This value is a bit low at less than one bit/second. This is expected since we only extract the sign of each sample and will be improved by sigma-delta modulation next.

With sigma-delta modulation in Table 4 the minimum entropy $\mathsf{H}_{\min}$ is around one bit per byte, but the number of samples increases 40 times and the entropy extracted each second H increases to 10-15 bits/second which is in-line with results from related work. The increase in the number of samples is easily explained by the fact that we choose $\delta = 10\ ms$ and since the sampling rate of the accelerometer is 200 *ms* we have 20 values for each sample. Moreover, due to the ternary encoding 0, 1, -1 we get 2 bits for each sigma value and thus $2 \times 20 = 40$ values on each sample point. The small decrease in entropy for each byte is well compensated by the larger data pool.

With simple-scaling and high-pass the results with sigma-delta modulation give lower adversary advantages when compared to Table 3. While in general high-pass filtering gave marginally lower performances than simple scaling, it also removes the undesired situation for simple-scaling when the adversary advantage is equal to 1. Interestingly, this problem with simple scaling did not occur when sigma-delta modulation was not used. With smoothness, the chances for pairing are higher but the adversary success rate also increases which rules out this mechanism. Rather unexpected, in case of the bike environment with smoothness, pairing fails at $\ell/8$. This may be due to poor alignment of the data since otherwise smoothness outperforms the rest of the filters in terms of the success rate of the pairing.

A graphic summary for the probability of identical vectors in all experiments is provided in Figure 12. Clearly, applying the smoothness filters results in a higher probability for matching. The sigma-delta modulation also helps in this respect. The train and tram environments are also the best for matching the feature vectors but notably these should also be more exposed to adversaries that may harvest data from the same environment.

To avoid overloading the main body of the work, we defer the discussion on various sampling rates, car and train scenarios for Appendix A.

III. PROTOCOL AND RESULTS

We proceed to the description of the proposed protocol then discuss some computational results.

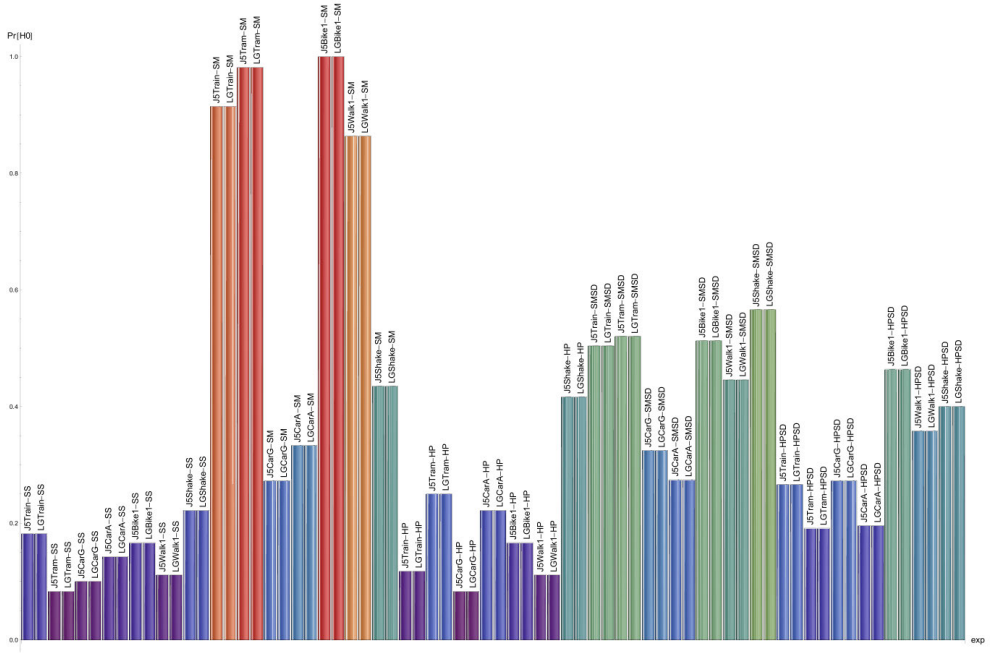

FIGURE 12. Probability of identical acceleration vectors for data recorded on LG Optimus and Samsung J5 in distinct environments with various filtering techniques.

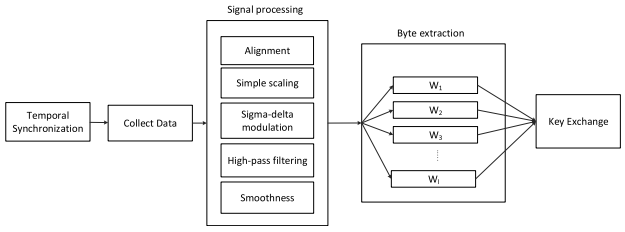

FIGURE 13. Flowchart for data processing and key-exchange.

A. EXCHANGING ACCELEROMETER DATA WITH EKE-DH

In Figure 13 we depict the flowchart of the key-exchange process starting from synchronization, collecting the data then processing and splitting it into ℓ windows for the final key-exchange. Figure 14 gives an outline of the EKE-DH based protocol. Multiple windows of accelerometer data are exchanged between the phones by Bluetooth connectivity. We give the formal description of the protocol next.

EKE-DH based protocol. We assume a large prime number p and generator g of Z_p fixed as public system parameters. Also, a parameter k is fixed as security level. Each of the two phones, A and B, follows the procedures below over wireless connectivity (Bluetooth in our experiments):

1) $\mathsf{Coll}(\Delta)$ in which both phones A and B collect data during a fixed time-windows Δ, apply the filtering algorithms (time-alignment, scaling, high-pass and sigma-delta modulation accordingly) then split the data into ℓ windows, i.e., $w_1^{\mathsf{id}}, w_2^{\mathsf{id}}, \ldots, w_\ell^{\mathsf{id}}$ where $\mathsf{id} \in \{A, B\}$;
2) $\mathsf{EKE-DH}(w_i^{\mathsf{id}}, \mathsf{id} \in \{A, B\}, i = 1..\ell)$ in which phones A and B exchange data using the Diffie-Hellman version of the EKE protocol by encrypting the Diffie-Hellman key-shares with the data from each window w, i.e.,

$$
\begin{aligned}
&for \quad i = 1..l \\
&\qquad A \rightarrow B : e_{w_1}(g^{a_1}) modp \\
&\qquad B \rightarrow A : e_{w_1}(g^{b_1}) modp, H(sk_1, 1) \\
&\qquad A \rightarrow B : H(sk_1, 2) \\
&\qquad \ldots \\
&\qquad A \rightarrow B : e_{w_\ell}(g^{a_\ell}) modp \\
&\qquad B \rightarrow A : e_{w_\ell}(g^{b_\ell}) modp, H(sk_\ell, 1) \\
&\qquad A \rightarrow B : H(sk_\ell, 2)
\end{aligned}
$$

where $sk_i, i = 1..l$ is the secretly shared Diffie-Hellman key, i.e., $sk_i = g^{a_i b_i} modp$, $i = 1..l$ and a_i, b_i, $i = 1..l$ are randomly generated and kept secret on each side;
3) $\mathsf{Extract}(\{(sk_1, H(sk_1, 1), H(sk_1, 2)), \ldots, (sk_\ell, H(sk_\ell, 1), H(sk_\ell, 2))\})$ where each principal A, B, validates the shares $sk_i, i = 1..\ell$ by checking $H(sk_i, 1)$ and $H(sk_i, 2), i = 1..\ell$ and keeps only the key shares for which the hashes are equal. The common session key is extracted via a key derivation function applied over the valid shares if and only if there are at least k valid key shares, otherwise the connection is closed.

In the previous description we considered that extraction occurs for ℓ samples. This can be repeated as soon as the transportation environment is changed (this can be detected both by changes in accelerometer patterns as well as by changes in speed). A ratcheting algorithm that accounts for previous key shares between devices can be used to increase

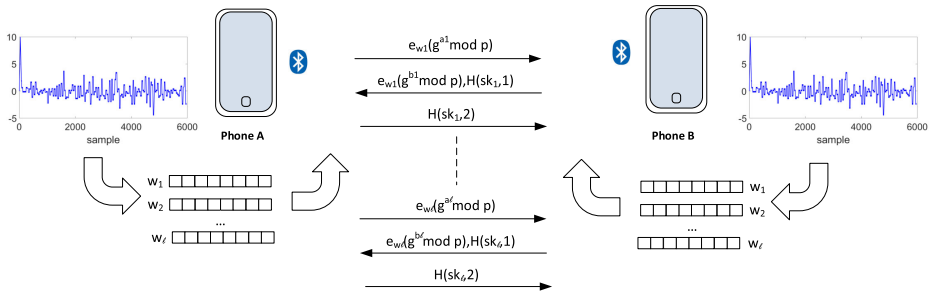

FIGURE 14. Data extraction and key-exchange between two phones.

TABLE 5. Computational time for the pairing operation, i.e., extracting one window of shared data, based on EKE-DH and SPEKE.

	EKE-DH Z_p-1024 bit		EKE-DH Z_p-2048 bit		SPEKE-192 bit (secp192r1)		SPEKE-256 bit (secp256r1)	
Phone	Share	Recover	Share	Recover	Share	Recover	Share	Recover
LG Optimus	$42ms$	$64ms$	$230ms$	$411ms$	$94ms$	$39ms$	$145ms$	$70ms$
Samsung J5	$25ms$	$40ms$	$138ms$	$248ms$	$21ms$	$7ms$	$37ms$	$11ms$
Samsung A3	$22ms$	$39ms$	$126ms$	$236ms$	$20ms$	$7ms$	$34ms$	$12ms$

the security of future sessions. This can be straight-forwardly done by including previous keys in the key derivation process for the current session key (cf. [7]).

B. COMPUTATIONAL RESULTS

A proof-of-concept pairing application was implemented on our Android smartphones. Table 5 holds computational results for the EKE-DH key exchange as we implement it in Java by using the Spongy Castle library. In case of the elliptical curve version of the EKE-DH protocol, decryption of the key shares, i.e., $e_w(aP)$ and $e_w(bP)$, may result in points that do not belong to the curve. This leads to entropy leakage since the correct key may lead only points from the curve (this is a known problem when using elliptical curves on EKE-DH). To fix it, we rely on the similar SPEKE protocol from Jablon [15] where rather then encrypting the point P with the secret, the point P is generated by using the shared secret as seed (in our case w). Recent work in [13] has shown some vulnerabilities of the protocol (impersonation under parallel sessions and key malleability), but the protocol can be easily fixed as pointed out in the same work.

In Table 5, we separate between the time needed to compute the key share by each participant, e.g., g^a or aP, and extracting the session key, e.g., g^{ab} or abP. The computational time is reasonable for both the 1024 and 2048-bit modulus in case of Z_p ranging from a 20 to around 200 ms. With elliptical curves the representation of the data is more compact and the computational time somewhat lower at key recovery, i.e., in the range of 7 to 69 ms. The advantages of elliptical curves do not appear significant when generating the shares (in fact on one of the phones this is even slower). This is because in each share we need to use the signal from the accelerometer w to generate a new base point P on the curve. To achieve this, we seed a pseudo-random number generator (PRNG) with accelerometer data w and XOR this with the X-coordinate of a regular point from the curve. Then, based on the X-coordinate we extract the point by computing the corresponding Y-coordinate. Note however, that not all X coordinates will belong to points on the curve since x^3+ax+b must be a square. So in case that extraction fails, we take the next random bytes from the PRNG to extract a new point and so on. This requires somewhat longer time to build the key share, i.e., aP, due to the extraction of a new P. In the Spongy Castle based implementation that we used, the advantages of using elliptical curves instead of the regular Z_p are small and become relevant only when extracting the session key or when it comes to the size of the key shares but not in the initialization step.

Our application extracts accelerometer samples based on the triggered event from the sensor, i.e., *onSensorChanged*, at 200 ms. As stated, in the theoretical analysis, the sample accounts for the accelerations along the X, Y, Z axes. This sample rate was the default and we did not increase it in order to avoid battery exhaustion but also to cope with the computational demands of the Diffie-Hellman protocol. At 200 ms sampling rate, the 56 samples of 8-bits collected in 90 seconds would require about 5.6 seconds, i.e., 56×100 ms, for exchanging the key with EKE-DH on our worst performer phone (LG Optimus). The key-exchange can be also done on the fly at the time when the samples are collected. We tried this and didn't note any performance degradation for the matching accuracy. If sigma-delta modulation is used, the number of samples increases significantly and it may be the case that there are too many samples to exchange by the Diffie-Hellman protocol. Selecting only a smaller number of samples can solve this problem and the success rate will remain similar.

The Android application validates the results from Tables 3 and 4. Concretely, similar numbers of matching vectors were achieved as in our theoretical analysis from Matlab and Mathematica.

IV. CONCLUSION

Accelerometer patterns differ significantly with transportation modes. Our results show that under all transportation environments accelerometer data carries sufficient entropy

TABLE 6. Results with simple scaling (SS) in various environments at 20ms and 100ms (Samsung A3).

Proc.	Env.	$p_{Adv\mathcal{D}}$	$p_{Adv\mathcal{H}}$	p_{Usr}	$\epsilon^{\ell/8}_{Adv\mathcal{D}}$	$\epsilon^{\ell/8}_{Adv\mathcal{H}}$	$\epsilon^{\ell/8}_{Usr}$	$\epsilon^{\ell/4}_{Adv\mathcal{D}}$	$\epsilon^{\ell/4}_{Adv\mathcal{H}}$	$\epsilon^{\ell/4}_{Usr}$	H_{min}	H	μ(Ham)	$\lvert\bar{v}_0\rvert$	ℓ
20ms	Tram	0.02	0.03	0.27	$< 10^{-31}$	$< 10^{-20}$	1.00	$< 10^{-102}$	$< 10^{-77}$	0.84	2.98	0.05	1.45	150	562
20ms	Bike	0.23	0.23	0.40	1.00	1.00	1.00	0.12	0.12	1.00	0.80	0.02	1.22	223	562
20ms	Walk	0.11	0.11	0.36	0.10	0.13	1.00	$< 10^{-21}$	$< 10^{-20}$	1.00	1.71	0.04	1.09	200	562
20ms	Shake	0.30	0.30	0.53	1.00	1.00	1.00	1.00	1.00	1.00	0.84	0.02	1.04	298	562
100ms	Tram	0.00	0.01	0.13	0.00	$< 10^{-12}$	0.54	0.00	$< 10^{-31}$	$< 10^{-4}$	3.81	0.04	1.87	14	112
100ms	Bike	0.07	0.07	0.45	0.03	0.03	1.00	$< 10^{-8}$	$< 10^{-8}$	1.00	2.64	0.04	0.96	50	112
100ms	Walk	0.05	0.05	0.24	0.00	0.00	1.00	$< 10^{-11}$	$< 10^{-11}$	0.45	2.17	0.04	1.27	27	112
100ms	Shake	0.01	0.04	0.22	$< 10^{-12}$	$< 10^{-3}$	1.00	$< 10^{-31}$	$< 10^{-13}$	0.28	2.32	0.04	1.08	25	112

TABLE 7. Results with simple scaling (SS) on experimental data inside the car at 20ms, 100ms and 200ms (Samsung J5).

Proc.	Env.	$p_{Adv\mathcal{D}}$	$p_{Adv\mathcal{H}}$	p_{Usr}	$\epsilon^{\ell/32}_{Adv\mathcal{D}}$	$\epsilon^{\ell/32}_{Adv\mathcal{H}}$	$\epsilon^{\ell/32}_{Usr}$	$\epsilon^{\ell/8}_{Adv\mathcal{D}}$	$\epsilon^{\ell/8}_{Adv\mathcal{H}}$	$\epsilon^{\ell/8}_{Usr}$	H_{min}	H	μ(Ham)	$\lvert\bar{v}_0\rvert$	ℓ
20ms	CarG	0.00	0.00	0.01	0.00	$< 10^{-10}$	0.00	0.00	$< 10^{-82}$	$< 10^{-42}$	2.00	0.03	3.15	8	562
20ms	CarA	0.01	0.02	0.05	0.00	0.05	0.99	$< 10^{-42}$	$< 10^{-33}$	$< 10^{-11}$	1.40	0.03	2.63	29	562
20ms	CarPA	0.02	0.02	0.09	0.03	0.05	1.00	$< 10^{-36}$	$< 10^{-33}$	0.01	2.27	0.04	2.30	53	562
20ms	CarGA	0.02	0.02	0.07	0.24	0.24	1.00	$< 10^{-27}$	$< 10^{-27}$	$< 10^{-5}$	1.55	0.03	2.72	41	562
100ms	CarG	0.03	0.03	0.07	0.58	0.58	0.99	$< 10^{-6}$	$< 10^{-6}$	0.03	1.42	0.03	2.85	8	112
100ms	CarA	0.03	0.03	0.05	0.58	0.58	0.94	$< 10^{-6}$	$< 10^{-6}$	0.00	1.00	0.02	2.63	6	112
100ms	CarPA	0.00	0.02	0.15	0.00	0.32	1.00	0.00	$< 10^{-8}$	0.82	3.09	0.04	1.69	17	112
100ms	CarGA	0.03	0.03	0.06	0.58	0.58	0.97	$< 10^{-6}$	$< 10^{-6}$	0.01	1.22	0.02	2.51	7	112
200ms	CarG	0.00	0.02	0.16	0.00	0.64	1.00	0.00	$< 10^{-4}$	0.82	3.17	0.04	1.84	9	56
200ms	CarA	0.02	0.02	0.05	0.64	0.64	0.95	$< 10^{-4}$	$< 10^{-4}$	0.03	1.58	0.02	2.88	3	56
200ms	CarPA	0.07	0.07	0.20	0.98	0.98	1.00	0.10	0.10	0.94	1.46	0.03	1.77	11	56
200ms	CarGA	0.07	0.07	0.11	0.98	0.98	1.00	0.10	0.10	0.39	0.58	0.01	2.48	6	56

for generating a secure session key. However, specific parameters, e.g., the length of the extraction sequence, must be tuned according to each scenario. Exchanging low-entropy values in a secure manner is possible due to guessing-resilience protocols that allow exchanging such values without exposing them to a brute-force search by a malicious adversary. The results that we obtain with various filtering techniques are somewhat close but even small differences may favour one or another technique. Due to its simplicity, simple scaling of the accelerometer values seems to be the best option. But sigma-delta modulation is beneficial for expanding the feature vectors and possibly increasing the entropy due to a better resolution, although one should consider that it also leads to more features to be exchanged and thus more computations. As differences between transportation environments exist, setting specific parameters according to the environment is necessary for a correct trade-off between the security level and pairing probability. By addressing both the pairing success rate and the adversary advantage we hope that we brought a crisper image in this direction.

APPENDIX A - RESULTS AT VARIOUS SAMPLING RATES AND SCENARIOS FOR CARS AND TRAINS

In the main body of the work we used a slow 200ms sampling rate since this was the default and all phones could easily cope with it. Clearly, a higher sampling rate will improve the results but it will also drain more battery and require more computations. Also, the car and train scenario require more attention for the specific placements of the phones. We discuss all these in what follows.

Results at various sampling rates. The LG Optimus did not behave well when we increased the sampling rate, most of the samples had significant clock drifts from the Samsung J5 which made pairing difficult. Therefore, we replaced the LG with a Samsung A3 and tried various pairing scenarios as shown in Table 6. The results at 20ms or 100ms do not show many improvements for most of the environments. The only significant improvement is in the case of the shaking process where p_{Usr} increases to 0.53 at 20*ms*. Perhaps this is to be expected since during shaking variations occur at a faster rate than when walking or going by train/tram. There is also a downturn in the car scenario where p_{Usr} but we discuss on this in what follows along with other experiments for cars and trains.

Experimenting with various car and train scenarios. While the J5 and A3 can cope with a higher sampling rate, the experimental data inside the car resulted in less vectors that match than in the rest of the scenarios, i.e, $p_{Usr} \in [0.1, 0.2]$. Consequently, we experimented with various placements for the phones inside the car. Four scenarios were tried: both phones in the glove compartment, both phones in the anti-slip, one phone in the anti-slip and the other in the glove compartment, one phone in the pocket of the passenger and the other in the glove compartment.

Table 7 summarizes the results at various sampling rates: 20 *ms*, 100 *ms*, 200 *ms*. The results generally lead to a p_{Usr} between 0.05 and 0.20 with simple scaling and we were unable to make a clear cut between the placements which suggests that speed and road obstacles (road bumpers, turning points, etc.) were more critical than the placement of the phone with our approach. The sigma-delta modulation again improved on the number of vectors that match. For brevity, in Table 8 we include results with sigma-delta modulation only for acquisition delays of 20ms and 200ms. While the data inside the car seems to have less entropy, it is still enough to make a clear cut from adversarial behaviour. The number of successful trials should be reduced to $\ell/16 - \ell/32$ as suggested in Tables 7 and 8 (in contrast to $\ell/4 - \ell/8$ as suggested in Table 6 for the rest of environments).

To clarify how these results relate to data extracted from distinct cars that are on the same road, we also add a car

TABLE 8. Results with sigma-delta modulation over simple-scaling (SSSD) on experimental data inside the car at 20ms and 200ms (Samsung A3).

Proc.	Env.	$p_{Adv\mathcal{D}}$	$p_{Adv\mathcal{H}}$	p_{Usr}	$\epsilon^{\ell/16}_{Adv\mathcal{D}}$	$\epsilon^{\ell/16}_{Adv\mathcal{H}}$	$\epsilon^{\ell/16}_{Usr}$	$\epsilon^{\ell/4}_{Adv\mathcal{D}}$	$\epsilon^{\ell/4}_{Adv\mathcal{H}}$	$\epsilon^{\ell/4}_{Usr}$	H_{min}	H	μ(Ham)	$\lvert\bar{v}_0\rvert$	ℓ
20ms	CarG	0.00	0.04	0.10	$< 10^{-147}$	$< 10^{-7}$	1.00	$< 10^{-945}$	$< 10^{-273}$	$< 10^{-96}$	1.32	0.03	2.10	219	2249
20ms	CarA	0.04	0.04	0.10	$< 10^{-5}$	$< 10^{-5}$	1.00	$< 10^{-259}$	$< 10^{-259}$	$< 10^{-94}$	1.23	0.02	2.17	221	2249
20ms	CarPA	0.05	0.05	0.14	0.00	0.00	1.00	$< 10^{-228}$	$< 10^{-228}$	$< 10^{-43}$	1.53	0.03	2.09	315	2249
20ms	CarGA	0.05	0.05	0.10	0.00	0.00	1.00	$< 10^{-224}$	$< 10^{-224}$	$< 10^{-87}$	1.06	0.02	2.30	231	2249
200ms	CarG	0.12	0.12	0.33	1.00	1.00	1.00	$< 10^{-68}$	$< 10^{-68}$	1.00	1.49	0.03	2.80	731	2245
200ms	CarA	0.10	0.10	0.23	1.00	1.00	1.00	$< 10^{-95}$	$< 10^{-95}$	0.03	1.26	0.03	3.15	523	2245
200ms	CarPA	0.15	0.15	0.36	1.00	1.00	1.00	$< 10^{-36}$	$< 10^{-36}$	1.00	1.27	0.03	2.58	800	2245
200ms	CarGA	0.14	0.14	0.29	1.00	1.00	1.00	$< 10^{-47}$	$< 10^{-47}$	1.00	1.12	0.03	3.01	660	2245

TABLE 9. Results with simple scaling (SS) in the car following scenario honda vs golf at 20ms, 100ms and 200ms (Samsung J5).

Proc.	Env.	$p_{Adv\mathcal{D}}$	$p_{Adv\mathcal{H}}$	p_{Usr}	$\epsilon^{\ell/16}_{Adv\mathcal{D}}$	$\epsilon^{\ell/16}_{Adv\mathcal{H}}$	$\epsilon^{\ell/16}_{Usr}$	$\epsilon^{\ell/4}_{Adv\mathcal{D}}$	$\epsilon^{\ell/4}_{Adv\mathcal{H}}$	$\epsilon^{\ell/4}_{Usr}$	H_{min}	H	μ(Ham)	$\lvert\bar{v}_0\rvert$	ℓ
20ms	CarF	0.00	0.00	0.01	$< 10^{-15}$	$< 10^{-15}$	$< 10^{-6}$	$< 10^{-102}$	$< 10^{-102}$	$< 10^{-61}$	2.00	0.02	3.98	4	562
100ms	CarF	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	4.08	0	112
200ms	CarF	0.02	0.02	0.02	0.64	0.64	0.64	$< 10^{-4}$	$< 10^{-4}$	$< 10^{-4}$	0.00	0.00	3.96	1	56

TABLE 10. Results with simple scaling (SS) on various placements of the LG optimus inside the train.

Proc.	Env.	$p_{Adv\mathcal{D}}$	$p_{Adv\mathcal{H}}$	p_{Usr}	$\epsilon^{\ell/8}_{Adv\mathcal{D}}$	$\epsilon^{\ell/8}_{Adv\mathcal{H}}$	$\epsilon^{\ell/8}_{Usr}$	$\epsilon^{\ell/4}_{Adv\mathcal{D}}$	$\epsilon^{\ell/4}_{Adv\mathcal{H}}$	$\epsilon^{\ell/4}_{Usr}$	H_{min}	H	μ(Ham)	$\lvert\bar{v}_0\rvert$	ℓ
200ms	Table-Arm	0.11	0.11	0.38	0.39	0.39	1.00	$< 10^{-3}$	$< 10^{-3}$	0.98	1.81	0.03	1.41	21	56
200ms	Table-Seat	0.00	0.04	0.13	0.00	$< 10^{-3}$	0.56	0.00	$< 10^{-9}$	$< 10^{-3}$	1.81	0.03	2.66	7	56

following scenario. Such a scenario also partly answers to the case of an external adversary. In this line of tests, a Honda Civic was following the VW Golf and the Samsung J5 was placed inside the Honda while the Samsung A3 is inside the Golf. The mobiles were placed in the anti-slip devices for both cars. In the car following scenario, as shown in Table 9, p_{Usr} is between 0 and 0.02 which shows that phones from distinct cars do not successfully pair. Since one car was a few meters behind, road obstacles were not reached at the same time and perhaps by proper time alignment the results would be better. A more careful analysis may be future work. Our intention here was only to prove that data collected from other cars does not directly lead to the same results.

Another experiment to which we endeavored was to try various placements of the phone inside the train. We tried two placements: one phone was placed on the table and the other either on the armrest near the table or on the seat nearby. This also partly answer to the case when both devices are carried by the same person (thus as closer to each other) or perhaps one of the devices belong to another passenger (or an adversary) and the devices are more distant to each other (i.e., distinct seats). The results are summarized in Table 10, for brevity we only give results for simple scaling. Indeed, for the case when the devices were closer, we had $p_{Usr} = 0.38$ while for the more distant placement we have $p_{Usr} = 0.13$. This supports the idea that devices belonging to the same user can be separated to form a private network while pairing with devices from other passengers is still possible at a lower p_{Usr}.

REFERENCES

[1] M. Bellare, D. Pointcheval, and P. Rogaway, "Authenticated key exchange secure against dictionary attacks," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 2000, pp. 139–155.
[2] S. M. Bellovin and M. Merritt, "Encrypted key exchange: Password-based protocols secure against dictionary attacks," in *Proc. IEEE Comput. Soc. Symp. Res. Secur. Privacy*, May 1992, pp. 72–84.
[3] D. Bichler, G. Stromberg, M. Huemer, and M. Löw, "Key generation based on acceleration data of shaking processes," in *Proc. Int. Conf. Ubiquitous Comput.* Berlin, Germany: Springer, 2007, pp. 304–317.
[4] M. Cao, L. Wang, H. Xu, D. Chen, C. Lou, N. Zhang, Y. Zhu, and Z. Qin, "Sec–D2D: A secure and lightweight D2D communication system with multiple sensors," *IEEE Access*, vol. 7, pp. 33759–33770, 2019.
[5] K. Chen, M. Lu, X. Fan, M. Wei, and J. Wu, "Road condition monitoring using on-board three-axis accelerometer and GPS sensor," in *Proc. 6th Int. ICST Conf. Commun. Netw. China (CHINACOM)*, Aug. 2011.
[6] M. K. Chong, R. Mayrhofer, and H. Gellersen, "A survey of user interaction for spontaneous device association," *CSURACM Comput. Surv.*, vol. 47, no. 1, pp. 1–40, May 2014.
[7] K. Cohn-Gordon, C. Cremers, B. Dowling, L. Garratt, and D. Stebila, "A formal security analysis of the signal messaging protocol," in *Proc. IEEE Eur. Symp. Secur. Privacy (EuroS&P)*, 2017, pp. 451–466.
[8] S. Dey, N. Roy, W. Xu, R. R. Choudhury, and S. Nelakuditi, "AccelPrint: Imperfections of accelerometers make smartphones trackable," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2014.
[9] D. Dolev and A. Yao, "On the security of public key protocols," *IEEE Trans. Inf. Theory*, vol. IT-29, no. 2, pp. 198–208, Mar. 1983.
[10] T. Feng and H. J. Timmermans, "Transportation mode recognition using GPS and accelerometer data," *Transp. Res. C, Emerg. Technol.*, vol. 37, pp. 118–130, Dec. 2013.
[11] M. Fomichev, F. Álvarez, D. Steinmetzer, P. Gardner-Stephen, and M. Hollick, "Survey and systematization of secure device pairing," *IEEE Commun. Surv. Tuts.*, vol. 20, no. 1, pp. 517–550, 1st Quart., 2018.
[12] B. Groza and R. Mayrhofer, "SAPHE: Simple accelerometer based wireless pairing with heuristic trees," in *Proc. 10th Int. Conf. Adv. Mobile Comput. Multimedia-MoMM*, 2012, pp. 161–168.
[13] F. Hao and S. F. Shahandashti, "The SPEKE protocol revisited," in *Proc. Int. Conf. Res. Secur. Standardisation*. Cham, Switzerland: Springer, 2014, pp. 26–38.
[14] Z. He, J. Cao, X. Liu, and S. Tang, "Who sits where? Infrastructure-free in-vehicle cooperative positioning via smartphones," *Sensors*, vol. 14, no. 7, pp. 11605–11628, Jun. 2014.
[15] D. P. Jablon, "Extended password key exchange protocols immune to dictionary attack," in *Proc. IEEE 6th Workshop Enabling Technol., Infrastruct. Collaborative Enterprises*, Jun. 1997, pp. 248–255.
[16] Q. Jiang, X. Huang, N. Zhang, K. Zhang, X. Ma, and J. Ma, "Shake to communicate: Secure handshake acceleration-based pairing mechanism for wrist worn devices," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 5618–5630, Jun. 2019.
[17] D. Kirovski, M. Sinclair, and D. Wilson, "The martini synch: Using accelerometers for device pairing," Microsoft Res., Redmond, WA, USA, Tech. Rep. MSR-TR-2007-123, 2007.
[18] J.-G. Krieg, G. Jakllari, H. Toma, and A.-L. Beylot, "Unlocking the smartphone's sensors for smart city parking," *Pervasive Mobile Comput.*, vol. 43, pp. 78–95, Jan. 2018.
[19] A. Kumar, N. Saxena, G. Tsudik, and E. Uzun, "A comparative study of secure device pairing methods," *Pervasive Mobile Comput.*, vol. 5, no. 6, pp. 734–749, Dec. 2009.

[20] J. Lester, B. Hannaford, and G. Borriello, "Are You with Me?'-Using accelerometers to determine if two devices are carried by the same person," in *Proc. Int. Conf. Pervasive Comput.* Berlin, Germany: Springer, 2004, pp. 33–50.
[21] Y. Liu and J. Niu, "Overlapped-shaking: A local authentication method for mobile applications," in *Proc. IEEE Comput., Commun. IT Appl. Conf. (ComComAp)*, Oct. 2014, pp. 93–97.
[22] Y. Ma, Z. Zhang, S. Chen, Y. Yu, and K. Tang, "A comparative study of aggressive driving behavior recognition algorithms based on vehicle motion data," *IEEE Access*, vol. 7, pp. 8028–8038, 2019.
[23] R. Mayrhofer, "The candidate key protocol for generating secret shared keys from similar sensor data streams," in *Proc. Eur. Workshop Secur. Ad-Hoc Sensor Netw.* Berlin, Germany: Springer, 2007, pp. 1–15.
[24] R. Mayrhofer, Fub, and I. Ion, "UACAP: A unified auxiliary channel authentication protocol," *IEEE Trans. Mobile Comput.*, vol. 12, no. 4, pp. 710–721, Apr. 2013.
[25] R. Mayrhofer and H. Gellersen, "Shake well before use: Intuitive and secure pairing of mobile devices," *IEEE Trans. Mobile Comput.*, vol. 8, no. 6, pp. 792–806, Jun. 2009.
[26] A. Mednis, G. Strazdins, R. Zviedris, G. Kanonirs, and L. Selavo, "Real time pothole detection using Android smartphones with accelerometers," in *Proc. Int. Conf. Distrib. Comput. Sensor Syst. Workshops (DCOSS)*, Jun. 2011, pp. 1–6.
[27] M. Muaaz and R. Mayrhofer, "Accelerometer based gait recognition using adapted Gaussian mixture models," in *Proc. 14th Int. Conf. Adv. Mobile Comput. Multi Media-MoMM*, 2016, pp. 288–291.
[28] V. M. Patel, R. Chellappa, D. Chandra, and B. Barbello, "Continuous user authentication on mobile devices: Recent progress and remaining challenges," *IEEE Signal Process. Mag.*, vol. 33, no. 4, pp. 49–61, Jul. 2016.
[29] Y. Qin, H. Luo, F. Zhao, C. Wang, J. Wang, and Y. Zhang, "Toward transportation mode recognition using deep convolutional and long short-term memory recurrent neural networks," *IEEE Access*, vol. 7, pp. 142353–142367, 2019.
[30] N. Ravi, N. Dandekar, P. Mysore, and M. L. Littman, "Activity recognition from accelerometer data," *Assoc. Advancement Artif. Intell.*, vol. 5, pp. 1541–1546, Jul. 2005.
[31] S. Reddy, M. Mun, J. Burke, D. Estrin, M. Hansen, and M. Srivastava, "Using mobile phones to determine transportation modes," *ACM Trans. Sensor Netw. (TOSN)*, vol. 6, no. 2, pp. 1–27, Feb. 2010.
[32] G. Revadigar, C. Javali, W. Xu, A. V. Vasilakos, W. Hu, and S. Jha, "Accelerometer and fuzzy vault–based secure group key generation and sharing protocol for smart wearables," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 10, pp. 2467–2482, Oct. 2017.
[33] D. Schürmann, A. Brüsch, N. Nguyen, S. Sigg, and L. Wolf, "Moves like jagger: Exploiting variations in instantaneous gait for spontaneous device pairing," *Pervasive Mobile Comput.*, vol. 47, pp. 1–12, Jul. 2018.
[34] V. Shoup, *A Computational Introduction to Number Theory and Algebra*. Cambridge, U.K.: Cambridge Univ. Press, 2009.
[35] P. Singh, N. Juneja, and S. Kapoor, "Using mobile phone sensors to detect driving behavior," in *Proc. 3rd ACM Symp. Comput. Develop.-ACM DEV*, 2013, p. 53.
[36] A. Studer, T. Passaro, and L. Bauer, "Don't bump, shake on it: The exploitation of a popular accelerometer-based smart phone exchange and its secure replacement," in *Proc. 27th Annu. Comput. Secur. Appl. Conf.-ACSAC*, 2011, pp. 333–342.
[37] F. Sun, C. Mao, X. Fan, and Y. Li, "Accelerometer-based speed-adaptive gait authentication method for wearable IoT devices," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 820–830, Feb. 2019.
[38] P. S. Teh, N. Zhang, A. B. J. Teoh, and K. Chen, "A survey on touch dynamics authentication in mobile devices," *Comput. Secur.*, vol. 59, pp. 210–235, Jun. 2016.
[39] L. Wang, H. Gjoreski, M. Ciliberto, S. Mekki, S. Valentin, and D. Roggen, "Enabling reproducible research in sensor-based transportation mode recognition with the Sussex-Huawei dataset," *IEEE Access*, vol. 7, pp. 10870–10891, 2019.
[40] S. Wang, C. Chen, and J. Ma, "Accelerometer based transportation mode recognition on mobile phones," in *Proc. Asia–Pacific Conf. Wearable Comput. Syst.*, 2010, pp. 44–46.
[41] J. Yu, Z. Chen, Y. Zhu, Y. Chen, L. Kong, and M. Li, "Fine-grained abnormal driving behaviors detection and identification with smartphones," *IEEE Trans. Mobile Comput.*, vol. 16, no. 8, pp. 2198–2212, Aug. 2017.
[42] H. Yüzugüzel, J. Niemi, S. Kiranyaz, M. Gabbouj, and T. Heinz, "ShakeMe: Key generation from shared motion," in *Proc. IEEE Int. Conf. Comput. Inf. Technol., Ubiquitous Comput. Commun.; Dependable, Autonomic Secure Comput.; Pervasive Intell. Comput.*, Oct. 2015, pp. 2130–2133.

BOGDAN GROZA received the Dipl.Ing. and Ph.D. degrees from Politehnica University of Timisoara (UPT), in 2004 and 2008, respectively, and the habilitation degree, in 2016, with a focus on the design of cryptographic security for automotive embedded devices and networks. He has been actively involved inside UPT with the development of laboratories by Continental Automotive and Vector Informatik. He is currently a Professor with UPT. Besides regular participation in national and international research projects in information security, he lead the Cseaman project, from 2015 to 2017, where he was also leads the Presence Project, from 2018 to 2020, a research program dedicated to the interaction between vehicles and smart-phones funded by the Romanian National Authority for Scientific Research and Innovation. As community services, he participated in over 30 conference program committees and regularly serves as reviewer for journals in the field.

ADRIANA BERDICH received the Dipl.Ing. and M.Sc. degrees from the Politehnica University of Timisoara (UPT), in 2017 and 2019, respectively, where she is currently pursuing the Ph.D. degree. From 2015 to 2018, she was a Software Developer in the automotive industry for Continental Corporation in Timisoara. She is also a research student in the Presence project focusing on environment-based device association and also continues as a software developer in the automotive industry for Vitesco Technologies focusing on power-train applications.

CAMIL JICHICI received the Dipl.Ing. and M.Sc. degrees from the Politehnica University of Timisoara (UPT), in 2016 and 2018, respectively, where is currently pursuing the Ph.D. degree. He works as a Young Researcher in the Presence project. His research interests are on the security of in-vehicle components and networks. He has been working as a software Integrator with automotive industry for Continental Corporation in Timisoara, since 2014.

RENÉ MAYRHOFER received the Dipl.Ing. (M.Sc.) and Dr. Techn. (Ph.D.) degrees from Johannes Kepler University Linz (JKU), Austria, and the Venia Docendi in applied computer science from the University of Vienna, Austria. He continues to be involved with Android platform security as a Domain Expert with JKU. He is currently the Head of the Institute of Networks and Security, JKU. Previously, he held a Full Professorship of mobile computing with the Upper Austria University of Applied Sciences, Campus Hagenberg, a Guest Professorship of mobile computing with the University of Vienna, and a Marie Curie Fellowship with Lancaster University, U.K. His research interests include computer security, mobile devices, network communication, and machine learning, which he currently brings together in his research on securing mobile devices and digital identity. Within the scope of u'smile, the Josef Ressel Center for User-friendly Secure Mobile Environments, his research group looked into full-stack security of mobile devices from hardware through firmware up to user interaction aspect. One particular outcome was a prototype for a privacy conscious Austrian mobile driving license (AmDL) on android smartphones supported by tamper-resistant hardware. He has contributed to over 80 peer-reviewed publications and is a reviewer for numerous journals and conferences.