Shake Well Before Use: Intuitive and Secure Pairing of Mobile Devices

Rene Mayrhofer and Hans Gellersen

Abstract—A challenge in facilitating spontaneous mobile interactions is to provide pairing methods that are both intuitive and secure. Simultaneous shaking is proposed as a novel and easy-to-use mechanism for pairing of small mobile devices. The underlying principle is to use common movement as a secret that the involved devices share for mutual authentication. We present two concrete methods, *ShaVe* and *ShaCK*, in which sensing and analysis of shaking movement is combined with cryptographic protocols for secure authentication. ShaVe is based on initial key exchange followed by exchange and comparison of sensor data for verification of key authenticity. ShaCK, in contrast, is based on matching features extracted from the sensor data to construct a cryptographic key. The classification algorithms used in our approach are shown to robustly separate simultaneous shaking of two devices from other concurrent movement of a pair of devices, with a false negative rate of under 12 percent. A user study confirms that the method is intuitive and easy to use, as users can shake devices in an arbitrary pattern.

Index Terms—Algorithm/protocol design and analysis, ubiquitous computing, mobile environments, authentication, human-centered computing, mobile applications.

1 Introduction

For mobile users, it is of potentially great value to associate a personal device with another mobile device in a spontaneous manner, without need for the involved devices to have prior knowledge of each other. Spontaneous associations can be for the purpose of short-lived interactions, for example, file transfer, synchronization and payment, or aimed at longer lived pairing, for example, of a device with an accessory. Wireless ad hoc networks provide for spontaneous discovery and interaction, but the wireless nature of the link gives rise to problems of authentication and security [1]. A first problem, resulting from the invisibility of the link, is for users to ascertain that their device associates with the intended target device, and not with another entity discovered on the network [2]. A second problem, due to the inherent susceptibility of wireless communication to attack, is the risk that a third party can gain control over the communication by establishing itself as man-in the-middle (MITM) between the user's device and the intended target [3].

Authentication of all potential peer devices via trusted third parties is impractical in a mobile computing world with pervasive interaction opportunities [4]. Instead, security needs to be bootstrapped directly between the involved devices, under the control of the user. This requires an out-of-band mechanism alongside the wireless channel, over which material for key generation and verification can be exchanged without risk of manipulation by an MITM. Any such mechanism must be human-assisted and have the user in the loop to control and/or verify that only the intended devices communicate over the out-of-band channel. An example of an out-of-band mechanism for peer device authentication is PIN code entry by the user into the involved devices [3].

A critical requirement for human-assisted device authentication is ease of use and alignment with the user's primary goal of establishing interaction with another device. Mechanisms such as key code entry can be acceptable for occasional use in situations that motivate explicit attention to security, but involve prohibitive overhead when users seek to engage in fast and short-lived interactions. The number of spontaneous interactions can be expected to grow significantly as mobile devices and wireless networks become ever more pervasive, and mechanisms for authentication of such interactions must involve only minimal user attention in order to scale for frequent use. An additional challenge is that wireless devices increasingly vary in form factor and user interface (UI). Specifically, smaller and more specialized mobile and wearable devices cannot be assumed to include key pads or displays—for instance, headsets as mobile phone accessory, wireless heart rate monitors, and mobile personal servers [5].

Authentication by shaking is a novel mechanism for human-assisted authentication and secure pairing of mobile devices. This new mechanism is defined by: the use of common movement of devices as shared secret for their mutual authentication; simultaneous shaking of devices as user method to effect common movement; and embedded sensing of device movement with a single accelerometer. Authentication by shaking gives users explicit control over pairing of devices with a simple gesture. Users are not required to learn and perform any specific shaking gesture, as the purpose of the gesture is to generate a one-time secret between the devices that are shaken together. Shaking

- R. Mayrhofer is with Vienna University, Dr.-Karl-Lueger-Ring 1, A-1010 Vienna, Austria. E-mail: rene@mayrhofer.eu.org.
- H. Gellersen is with the Computing Department, Lancaster University, InfoLab21 South Drive, Lancaster LA1 4WA, UK. E-mail: hwg@comp.lancs.ac.uk.

Manuscript received 20 July 2008; revised 21 Jan. 2009; accepted 10 Feb. 2009; published online 18 Feb. 2009.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-2008-07-0279. Digital Object Identifier no. 10.1109/TMC.2009.51.

affords user control while eliminating the need for an explicit user interface. Instead, the device-to-user interface is reduced to an acceleration sensor coupled with a minimal feedback channel (e.g., an LED or a beeper). This makes authentication by shaking useful for a wide range of mobile devices, from embedded "UI-less" devices such as sensor nodes, portable hard drives, and wireless accessories to mobile phones and handhelds, many of which are beginning to be routinely equipped with accelerometers.

We introduce two concrete methods that use shaking for authentication. The first method, *ShaVe (shaking for verification)*, is based on standard key agreement over an insecure wireless channel with subsequent encrypted exchange of accelerometer data. The devices use the sensor data to independently verify that key agreement has taken place with the intended target device, as identified by sufficiently similar movement. The second method, *ShaCK (shaking to construct a key)*, represents an alternative approach in which devices use accelerometer data to extract feature vectors as building blocks for construction of a key. The feature vectors are exchanged as candidate key material between the devices in a cryptographic protocol that results in a shared key only if there is a sufficient match of candidate material. The two methods highlight design trade-offs: ShaVe uses a conservative and better understood cryptographic design and provides stronger security; ShaCK is computationally less expensive and allows more dynamic interaction, for example, shorter shaking phases if less security is required.

This paper is organized as follows: In Section 2, we reflect on the problem area and related work to position our contribution. This is followed by an introduction to our concept and overview of the concrete methods we implemented (Section 3). The major components in our approach are sensor data preprocessing (Section 4) as a prerequisite for determining if sensor time series are sufficiently similar (Section 5) and cryptographic protocols for coupling secure channel keys with sensor data comparison (Section 6). The algorithms for sensor data analysis are evaluated in Section 7, followed by an evaluation on system level, including usability study, embedded implementation, and security analysis (Section 8).

2 Related Work

The problem of peer device authentication for spontaneous association of devices in wireless ad hoc networks was first highlighted by Stajano and Anderson, in work that focused on a security policy for spontaneous and transient associations [1]. The authentication and pairing mechanisms that have since been investigated can be grouped into three different approaches: human verification of target authenticity based on direct output of the involved devices; direct user input of authentication material into the involved devices; and user selection of target devices over physically constrained channels.

The first approach, human verification, can be thought of as interactive challenge-response protocol, with the user in the role of verifying device responses. Most state-of-the-art methods rely on visual output, for example, displaying of six-digit codes in the numeric comparison mode of Bluetooth Simple Pairing [6], words from a dictionary as in DH-SC [7], and "Random Art" visualizations of hashes proposed to make the user's comparison task easier and more robust [8]. The use of visual output limits the range of target devices to ones that reasonably include a screen component. However, it has also been proposed to use blinking patterns of LEDs for verification of device authenticity [9], [10]. An alternative is auditory device feedback, for example, speech generated by target devices in the "Loud and Clear" protocol [11], however, with the limitation that sound generation can be disruptive and inappropriate. In contrast to this range of methods, authentication by shaking is cognitively less demanding for the user as it does not involve any verification task.

In other approaches, the user is in the role of providing explicit input into the involved devices to ensure that the intended pair authenticates. A standard mechanism is manual transfer of PIN codes and passwords [3]; Bluetooth "passkey entry," for instance, requires the user to type a numeric code that is displayed on one device into the other device. An alternative to keypad entry are synchronized button presses that only require a single button for each device or a visual indicator on one and a button on the other [12]. It has also been suggested that instead of a code, one of the devices could display a gesture for the user to perform with the other device [13]. Our shaking-based mechanism can be regarded as a method relying on user input, but it differs fundamentally in not requiring the user to input any specific code or secret; instead, user input generated by shaking is random but simultaneous to the involved devices.

A range of methods have been proposed that employ physically limited channels for out-of-band communication between peer devices. In these methods, the user is in the role of physically controlling that only the intended devices are connected over such a channel. Balfanz et al. proposed *location-limited channels* based on transmission media with directed and limited propagation, and *demonstrative identification* of target devices by the user, for instance, by way of pointing their device [14]. Physically limited channels investigated for peer authentication include infrared beams [15], laser beams [16], [17], ultrasound [2], [18], using constraints such as distance bounding [7] and radio environment comparison [19]. Examples for physical selection and control by the user include pointing of their devices for target selection with laser or infrared [16], [17], [15], simultaneous button press on the involved devices [20], and use of a camera in their device to capture a visual code on the target device [21]. Our approach builds on the principal idea of limited channels and physical selection; it introduces a *movement-limited channel* (Section 3.1), and adopts shaking as a new user method for physical selection of devices and control over their authentication.

The idea of shaking two (or possibly more) devices together to pair them was first demonstrated in "Smart-Its Friends" [22]. The technique was based on broadcast of movement features over an insecure wireless channel and a simple heuristic for comparison of reported movements, without authentication of the involved devices. Closely related are "Synchronous Gestures" which have been proposed for setting up user interaction across devices, exemplified with "bumping" of one display device against another to form a shared larger display [23]. Accelerometer-based analysis of movement has also been investigated for context sensing in wearable and pervasive computing, for instance, to determine if devices are carried by the same person [24]; to group devices that are moving together [25]; and to identify device usage by correlation with user-worn sensors [26]. Of the movement analysis algorithms introduced in these contributions, we are adopting Lester et al.'s *coherence* measure [24] for comparing data time series in one of our implementations.

Shaking of devices for the purpose of using shared movement for authentication was originally introduced by us in a conference paper on which this paper expands [27], but has since been explored also by others. Kirovski et al. present a method in which devices that are moved together perform joint fuzzy hashing, similar in general design to the second of our two methods, ShaCK, but with differences in how key material is extracted from acceleration time series [28]. Bichler et al. likewise present an approach in which acceleration data is used for key generation, and in this sense, also more closely related to our second method, however using acceleration features in the time domain [29] (whereas both our methods are based on features in the frequency domain). Beyond these recent efforts, we contribute an implementation of two alternative methods to understand trade-offs in authentication based on accelerometer data. Last, we note that device shaking and rotating had also been proposed in prior work on pairing of CPU-constrained wireless devices, however restricted to the purpose of preventing that the involved devices are easily distinguishable by signal strength [30].

3 Authentication by Shaking

We introduce authentication by shaking as an approach to secure pairing that is designed to exploit joint movement of devices as shared secret. In this section, we discuss the design of our approach, with respect to three principal considerations: the use of shaking to create a movement-limited channel; the use of accelerometers for embedded sensing of movement (and the resulting need to reconcile inherent variance in sensor data streams); and the use of acceleration data in the authentication process, for key verification versus key generation.

3.1 Creating a Movement-Limited Channel

Balfanz et al. defined channels as *location-limited* if they have the property that human operators can precisely control which devices are communicating with each other, by way of using limited media and controlling the spatial arrangement of devices (e.g., direct pointing of devices at each other). Accordingly, we call a channel *movement-limited*, if it affords precise user control over which devices can communicate, by way of controlling their movement and by using movement as shared secret.

We propose simultaneous shaking of devices held together (in one hand) as concrete technique for creation of movement-limited channels, because shaking has characteristics on which we can build for easy-to-use and secure authentication:

- *Shaking is an intuitive gesture.* Humans are familiar with shaking as a form of physical interaction with objects that is very common, does not require learning, and is performed without cognitive demand; shaking is hence natural and easy-to-use as a user interaction technique [22].
- *Shaking motion is vigorous and distinctive.* Shaking involves pronounced accelerations over a longer duration than alternative gestures that could be performed with two devices, such as bumping together [23], allowing shaking to be separated robustly from other motion patterns.
- *Shaking movements are variant.* Shaking is a free-form gesture that naturally varies from one instance to another. We presume and experimentally validate (cf. Section 7) that this allows robust detection of whether a pair of devices are shaken together or shaken independently.

The mechanism we introduce is not prescriptive as to how users have to shake devices in order to authenticate. Specifically, users do not have to try and match any particular pattern or rhythm, because the shaking gesture is not used directly as a key but as a random shared secret. There is also no attempt to identify and authenticate the user on the basis of their shaking gesture; authentication is between devices, with mutual identification based on an ephemeral shared "shaking experience."

3.2 Using Acceleration Time Series

We base our authentication mechanism on localized sensing of movement with a single embedded accelerometer. This allows devices to independently capture movement, and specifically shaking, as a time series of acceleration values. Independent sampling of acceleration data is critical for the purpose of using movement as shared secret; any exchange of raw data would compromise this. A consequence of independent sampling however is that the time series are not aligned for comparison. We address this, as detailed in Section 4, with local preprocessing and normalization of the acceleration data prior to their use for authentication. The result of local processing are *active segments* of acceleration time series detected as representing shaking movement.

The separately measured movement of devices that are shaken together will be similar but not identical. This is partly due to the inherent imprecision of sensors but also due to the slightly different trajectories that devices can describe during shaking, dependent on how they are held together (e.g., think of one device being on the inner curve and the other one the outer curve of a shaking path). Authentication based on captured accelerometer data therefore needs to accommodate a degree of variance, rather than strict matching of time series. We do this in different ways in our two methods: in the ShaVe method, devices exchange acceleration data and then apply a similarity measure and threshold; in ShaCK, devices exchange variants of acceleration feature vectors and then use the percentage of matches found as a similarity measure.

3.3 Key Generation versus Key Verification

An out-of-band channel can be used in two different ways in the process of authenticating wireless devices. One way is to directly exchange key material out-of-band to guarantee

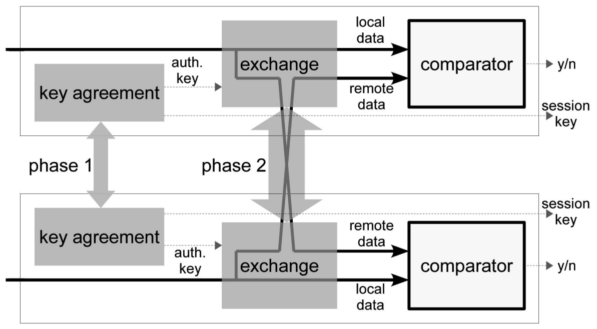


Fig. 1. In the ShaVe method, accelerometer data is used for key verification following key agreement over the wireless channel.

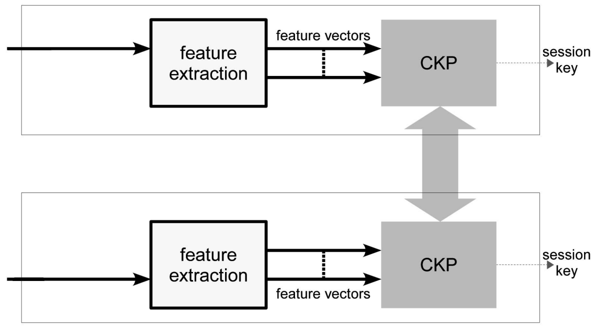


Fig. 2. In the ShaCK method, feature vectors are extracted from accelerometer data and directly used for construction of a shared key.

that key exchange occurs only between the intended devices. An example is manual transfer of keys or of codes from which keys are generated [3]. The other way is more indirect: having devices first agree keys over the insecure wireless channel and then using out-of-band communication to verify that the key has been exchanged with the intended party. Examples include verification over visual, auditory, and ultrasonic channels [8], [11], [18]. Direct key agreement out-of-band cuts out the need for subsequent verification, but the level of security can be limited by the source data. The key verification approach is more complex but the "strength" of the session keys is independent of the out-of-band mechanism.

Shaking-based sensor data can be used for both key generation and key verification. Acceleration data of vigorous shaking has, as we will show experimentally, sufficient entropy to provide material for robust and secure key generation. However, acceleration data is naturally constrained by user movement and measurement method; keys generated from the data will therefore not be as strong as ones resulting from key agreement over random numbers. Intuitively, there is a trade-off between use of sensor data for key generation as the more direct and presumably computationally less expensive approach, and its use for key verification following wireless key agreement as the more conservative approach from a security point of view. In order to explore this trade-off, we have designed and implemented both alternatives, in two concrete methods for shaking-based authentication.

The first method, ShaVe, is based on unauthenticated key agreement with subsequent verification. The method involves the following steps, as illustrated in Fig. 1:

1. generation of a secret key with Diffie-Hellman (DH) key agreement;
2. encrypted exchange of "raw" active segments of acceleration data via an Interlock protocol;
3. comparison of received data with locally captured active segments using a coherence measure;
4. if the check is successful then the agreed key is adopted as session key for secure communication.

In the second method, ShaCK, devices use accelerometer data directly for interactive key construction. As illustrated in Fig. 2, this involves the following:

1. continuous extraction of discrete feature vectors from active segments of acceleration data;
2. in each time step, a set of variant feature vectors are extracted as candidates for match-making;
3. exchange of feature vectors in an interactive cryptographic protocol that uses matching candidates as building blocks for a shared key;
4. if the protocol finds matching candidates at a sufficient rate, then this will directly result in a shared key after a number of iterations.

The two methods have in common that they build on preprocessing of accelerometer data in the time domain to detect and normalize active segments. In both methods, further analysis is conducted in the frequency domain, in ShaVe, for comparison of exchanged active segments, and in ShaCK, for extraction of feature vectors. Frequency domain processing has the advantage that it tends to be less sensitive to minor skew in compared time series, as well as to noise and offsets caused by uncalibrated sensors. In ShaVe, frequency domain analysis is done in one pass over a complete time series of acceleration values; in ShaCK, in contrast, acceleration data is continually transformed to produce a stream of feature vectors of equal bit length.

4 Preprocessing of Accelerometer Data

Accelerometer data is captured and locally preprocessed in three steps as shown in Fig. 3, to obtain normalized time series that can be used subsequently for analysis of whether devices have been shaken together.

4.1 Sensor Data Acquisition

Sensor data is captured as time series of acceleration values in three dimensions, sampled at equidistant time steps. The data must be captured independently by the involved devices; any leak of this raw data would compromise its use

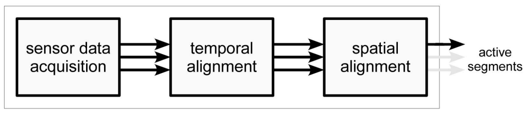


Fig. 3. Local preprocessing of accelerometer data: capture of time series in 3D, temporal alignment for segmentation, and spatial alignment for combination into a single normalized data stream.

for mutual authentication as it could be exploited for side-channel attacks [31].

Practical experience shows that sampling rates between 100 and 600 Hz are appropriate for capturing shaking. In our experiments, we sampled at 512 Hz and analyzed the impact of lower rates.

4.2 Segmentation for Temporal Alignment

As the devices sample accelerometer time series independently, temporal alignment is required to facilitate comparison. We use an event-based approach in which devices detect the onset of a shaking event, and use this to segment the data. Devices can detect the onset of shaking independently which means that there is no need for explicit synchronization between devices, assuming they are equipped with sufficiently accurate real-time clocks to ensure alignment within detected active segments. The detected onset of shaking also serves as trigger to start the authentication process, eliminating the need for any additional user input to indicate the intent to pair a set of devices.

The start of an active segment is detected when the variance of acceleration data within a sliding window exceeds a threshold. Empirically, best results for our purposes have been obtained at a sampling rate between $f = [128; 512]$ Hz with a sliding window of $v = f/2$ samples (i.e., 0.5 seconds), and a normalized variance threshold around $T_\sigma = 0.045$. The end of an active segment can be determined when the motion subsides (this approach is used in the ShaCK method) or can be defined by a fixed segment length (this is applied in ShaVe, with segments of 3 seconds duration).

4.3 Normalization for Spatial Alignment

How a user holds two devices together can vary, and independently recorded 3D acceleration time series will therefore typically lack spatial alignment and not be directly comparable. We address this by reducing the three dimensions to a single one, combining the acceleration values captured along the x, y, and z axes. A sample in the resultant data stream represents the magnitude of the movement vector while its direction is not used in the current implementation. This means that information of potential use for subsequent analyses is discarded but has the advantage of low cost implementation, in comparison with methods such as principal component analysis (PCA).

For combination of acceleration values across the three dimensions, and for subsequent comparability of data streams, raw samples are normalized to a range centered around 0, i.e., $[-a; a]$. We use a dynamic approach in which sensor data is analyzed in a sliding window, first scaling values from the sensor-specific range to $[0; 2a]$, and then subtracting the sliding average separately for each dimension before computing the magnitude.

5 Algorithms for Detection of Shared Movement

Independently captured and preprocessed time series are compared to determine if devices have been shaken while held together. This is a binary classification problem to separate the intentional "shaking together" of devices from any concurrent but spatially separated shaking of devices. We propose two alternative algorithms for classification, both based on the correlation of active segments in the frequency domain. The first algorithm produces a measure of coherence, to directly indicate how similar two given time series are, and is a building block for the ShaVe method (*shaking for verification*). The second algorithm introduces candidate feature extraction and measures similarity of shaking movements in terms of matching features found in the two time series. The first algorithm is focused on deciding if devices have been shaken together, whereas the second algorithm also produces candidate features suitable for key generation. This second algorithm is hence used in the ShaCK method (*shaking to construct a key*).

In both algorithms, the decision process needs to fulfill two conflicting aims: 1) to be robust to small variations due to slightly different device trajectories and sampling noise, and 2) to extract a sufficient amount of pseudo-randomness for use in the authentication protocol. For secure authentication, we require high entropy from an adversary's point of view, i.e., that any device that is not being shaken with the intended devices shall be sufficiently uncertain about those parts of the sensor data that are used for deciding on similarity.

5.1 Coherence of Acceleration Time Series

We adopt a coherence function described by Lester et al. in their work on using accelerometers to determine whether devices are carried together [24]. The coherence function is a measure of linear correlation of acceleration data in the frequency domain. For computation of coherence, two time series a and b are split into n (optionally overlapping) averaged slices a_k and b_k, normalized by the signal power spectra. The function is then approximated by the magnitude squared coherence (MSC):

$$C_{xy}(f) = \frac{P_{xy}(f)}{P_{xx}(f) \cdot P_{yy}(f)} \tag{1}$$

with (cross-) power spectra

$$P_{xy}(f) = \frac{1}{n}\sum_{k=0}^{n-1} x_k(f) \cdot \bar{y_k}(f) \tag{2}$$

computed over FFT coefficients $x_k(f) = FFT(a_k(t) \cdot h(t))$ and $y_k(f) = FFT(b_k(t) \cdot h(t))$ using the standard von-Hann window $h(t) = \frac{1-\cos(2\pi t/w)}{2}$.

Note that while the signals a and b in time domain are real, their FFT coefficients x and y are complex, and $\bar{x}$ refers to the conjugate complex of x. By using squared magnitudes, C_{xy} becomes real-valued. The significance of coherence values depends on the number of averaged slices n: the more slices, the lower the coherence values for the same pair of signals. Therefore, ShaVe uses a constant length of 3 seconds as a compromise between sufficient variability for robust classification respectively entropy from an adversary's point of view and quick user interaction. The final coherence value is computed by averaging up to a cutoff frequency f_{max}:

$$C_{xy} = \frac{1}{f_{max}}\int_0^{f_{max}} C_{xy}(f)df. \tag{3}$$

With this heuristic, C_{xy} is thresholded to decide whether devices have been shaken together. Separation of "shaking together" and "shaking independently" has been tested with a range of parameter settings, showing effective separation with a sampling rate of $R = 256$ Hz, windows of $W = 256$ samples with an overlap of 7/8, and a cutoff frequency of $F_{max} = 40$ Hz.

5.2 Candidate Feature Matching

The feature vectors on which the coherence algorithm operates are based on continuous-valued FFT coefficients. Even when devices are rigidly held together for shaking, the feature vectors will naturally exhibit a degree of variance. The principle underlying our second algorithm is to build feature vectors based on quantified FFT coefficients so that we can expect to find bit-equal matches of feature vectors as indicator for coherent movement.

Our candidate feature extraction algorithm is based on exponentially quantized FFT coefficients that are pairwise added, as suggested by Hyunh and Schiele [32], and empirically confirmed to be more effective than linear quantization and unpaired use of coefficients. First, the (nonaveraged) power spectrum is computed:

$$P_{xx_k}(f) = x_k(f) \cdot \bar{x_k}(f) \tag{4}$$

using FFT coefficients $x_k(f) = FFT(a_k(t))$ based on overlapping slices a_k of the signal a without applying specific windowing functions. These coefficients are pairwise added

$$Q_k(f) = P_{xx_k}(f) + P_{xx_k}(f + \delta_F), \tag{5}$$

where δ_F describes the frequency resolution after FFT, i.e., the difference between two frequency bands. When aiming for equivalence of feature vectors, there is however an additional complication: small differences of values near the boundaries of quantization bands can lead to different feature values, although the FFT coefficients are only marginally different. The proposed solution is to generate M *candidate* feature vectors with different offsets with respect to B exponentially scaled quantization levels:

$$\forall_{m:=0\ldots M-1}\; v_{km}(f) := b \text{ so that } l_{m_b} \leq Q_k(f) < l_{m_{b+1}}. \tag{6}$$

The quantization level boundaries l_{m_b} are computed as $l_{m_b} = (\sum_{i=0}^{b-1} 2^b + o_m \cdot 2^b) \cdot \frac{\max(Q_k)-\min(Q_k)}{B} + \min(Q_k)$ with offsets $o_m = \frac{m}{2(M-1)}$ ranging from 0 to the value of the smallest quantization band.

Given two acceleration time series, the similarity criteria used with this algorithm is the rate of matches in a stream of candidate feature vectors. Thresholding the rate at a certain percentage produces a binary decision for the authentication protocol. In our experiments, detailed further below, best results for distinguishing shaking together from shaking independently have been observed with $B = 6$ exponentially scaled quantization bands and $M = 4$ candidate vectors, at a sampling rate of $R = 512$ Hz with FFT windows of $W = 512$ (1/2 overlap) and a cutoff frequency of $F_{max} = 20$ Hz.

6 Authentication Protocols for Secure Pairing

We propose two different authentication protocols. The first one is based on the well-known Diffie-Hellman and Interlock protocols and used for a key verification approach in combination with coherence analysis of accelerometer data. The second one is a new protocol introduced for generating keys from candidate material extracted from sensor data streams, and in the context of this work, is combined with the candidate feature extraction from acceleration time series as described above.

6.1 Diffie-Hellman and Interlock* for Key Verification

In the ShaVe method, accelerometer data is used for key verification. The method adopts standard DH key agreement [33] followed by an exchange of preprocessed accelerometer data that are secured with an Interlock protocol.

Using DH key agreement over an insecure channel, devices A and B generate two shared keys K^{Auth} and K^{Sess}, one for authentication and the other as session key to provide forward secrecy. The keys are secure in the sense that it is impossible to infer one from the other but they are not authenticated. Key agreement is hence followed by a verification process to ascertain that the key is held by the intended party, and to rule out any potential MITM. The unique key property of DH guarantees that if $K_a^{Auth} = K_b^{Auth}$, there can be no adversary E with $K_{e1}^{Auth} = K_a^{Auth}$ and $K_{e2}^{Auth} = K_b^{Auth}$, and subsequently, no $K_{e1}^{Sess} = K_a^{Sess}$ and $K_{e2}^{Sess} = K_b^{Sess}$.

The verification is based on encrypted exchange of a shared secret via an Interlock protocol [34]. The essence of Interlock is that messages are split after block encryption, and that devices are forced to send a part of their message before they receive the next part, and thus, before they can decrypt the whole message block. Interlock therefore protects against an MITM: an adversary E between A and B will not be able to decrypt a partial message, and both forwarding of the original message without reencryption, and forwarding of a newly inserted message will lead to uncover E (in the first case, because A and B would not be able to decrypt the reassembled message due to different session keys, in the second case, because A and B would note the modified message content when the decoded sensor data streams are not sufficiently similar).

For use with sensor data streams as shared secret, we introduce *Interlock** as an extension of the original protocol. In this variant, A and B encrypt their complete messages, i.e., the (zero-padded) acceleration data vectors a and b with lengths of n and m blocks, respectively, with any of the well-known block cipher modes. To prevent simple mirroring attacks, the local device identifier (e.g., its unique Bluetooth MAC address) is perpended to a and b. In our implementation of ShaVe, cipher block chaining (CBC) mode with a random initialization vector (IV) is used. The resulting cipher texts c and d with lengths of $n + 1$ and $m + 1$ blocks are then split into two messages by concatenating the first halves of all cipher blocks into the first messages A_1 and B_1 and the second halves of all cipher blocks into the second messages A_2 and B_2. This ensures that an adversary E cannot decrypt any of the blocks, and is therefore barred from learning any parts of the plain text messages.

After exchange and reassembly of messages a and b, A and B each verify that the received identifier matches the remote device and the received acceleration data is similar to the locally captured data, i.e., that $a \sim b$. In our implementation of the ShaVe method, the coherence algorithm described above is used for this purpose. The protocol however is more general and can be combined with other criteria and algorithms for verifying sufficient similarity of exchanged sensor data. Both devices finally indicate success or failure of the verification to the other device over the wireless channel and signal the outcome to the user. User feedback can be minimal (e.g., green/red LED) but is required both for usability (transparency of the interaction state) and security (e.g., to signal failed key verification in the case of an attempted MITM attack).

6.2 Candidate Key Protocol

In the *candidate key protocol* (CKP), a shared secret key is generated directly from sensor data instead of by DH agreement. As presented in detail in Appendix A, feature vectors v are hashed to generate *candidate key parts* h. When feature extraction produces multiple "parallel" feature vectors v^i for each time window in the sensor data, then these serve as candidate key parts h^i. The one-way hashes are a simple way to communicate that a device has generated a certain feature vector without revealing it. To make dictionary attacks harder, the standard approach of perpending random salt values s before hashing is used. When B receives such candidate key parts from A, it can use its own history of recently generated feature vectors LH to check for equals. When B has generated the same feature vector, it is stored in a list of *matching key parts* MC specific to each communication partner. As soon as enough entropy has been collected in this list, B concatenates all feature vectors, appends a constant C, hashes the resulting string, and sends a *candidate key* K to A. If no messages have been lost in transit, A should be able to generate a key with the same hash, and thus, the same secret key, which it acknowledges to B. If messages have been lost, A can ignore a candidate key and create its own later on.

CKP is a general protocol that can be used with any feature vectors. Here, it is applied to quantized FFT coefficients, which work well for accelerometer data. A more thorough specification of CKP and a description of implementation issues has been presented in another publication [35]; a discussion of its security properties is provided in the security protocol analysis below.

7 Evaluation of the Classification Algorithms

An assumption underlying the concept of authentication by shaking is that it is possible to robustly detect whether a pair of devices have been shaken together, as opposed to shaken independently. To validate this assumption, we have collected multiuser data sets and evaluated the classification algorithms introduced in Section 5. We have used this data collection also to analyze the influence of a variety of parameters on algorithm performance.

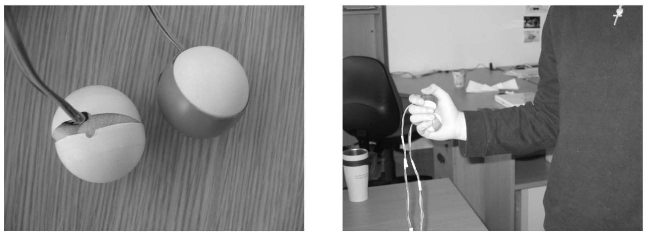


Fig. 4. Data collection was carried out with accelerometers mounted in ping-pong balls, to allow for arbitrary spatial alignment of the sensors.

7.1 Design and Procedure

The design of our experiment was based on collection of two data sets, representing the conditions *shaking together* versus *shaking independently*. The aim of the experiment was to test whether the algorithms introduced in Section 5 are able to robustly detect the "positive" condition *shaking together*. Both algorithms use thresholds for classification, in the ShaVe method based on a coherence measure, and in the ShaCK method based on a rate of matching candidate feature vectors. The test hypothesis was therefore that thresholds can be found for these values above which the algorithms do not produce any false positives (i.e., any misclassification of the "negative" condition *shaking independently* as positive). The experiment further aimed to identify parameter sets for the algorithms to minimize the false negative rate (i.e., misclassifications of *shaking together* as negative).

As apparatus for data collection, we mounted accelerometers in a pair of ping-pong balls, fixated with compressed foam (see Fig. 4). Ping-pong balls were chosen as representative in size for mobile devices and accessories, and because the ball shape did not constrain alignment; this allowed for collection of a data set in which pairs of accelerometers were randomly oriented with respect to each other. In each of the balls, two ADXL202JE accelerometers were fixed at an angle of 90 degree so that all three dimensions could be measured (the use of 2D accelerometers was a pragmatic choice). The accelerometers measured acceleration of up to 2 g in each dimension, and their pulse-width outputs were set to a sampling rate of about 600 Hz. Polling the outputs at around 1 MHz provided a resolution of about 10 bits per sample. The accelerometers were wired for the data collection but the attached cables were lightweight, flexible, and long enough so as not to disturb or limit the movement of devices by the subjects.

The data collection was carried out in two separate phases. In the first phase, data were collected from individual participants to obtain a data set for the *shaking together* condition. In the second phase, data were collected from pairs of participants shaking devices simultaneously but independently, to obtain data representing the "negative" condition.

For collection of the first data set, participants were asked to shake the two ping-pong balls together but were explicitly not instructed as to how to perform the shaking movement. For each subject, 30 trials of approximately

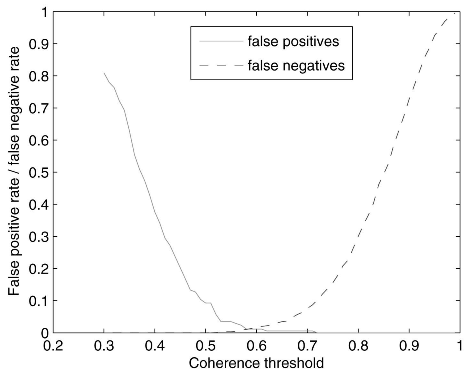


Fig. 5. The coherence-based algorithm used in the ShaVe method rejects false positives above a threshold of $C_{xy} = 0.72$, with a false negative rate of 10.24 percent at this threshold.

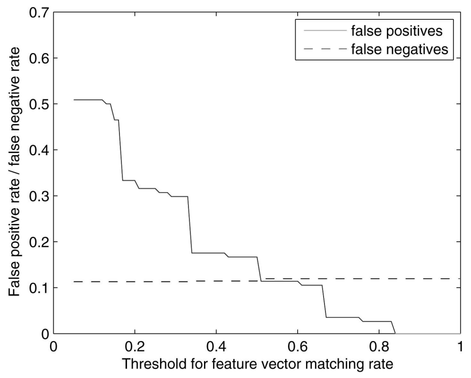


Fig. 6. The matching features algorithm used in the ShaCK method rejects false positives above a threshold of 0.88; the false negative rate is nearly constant with a maximum of 11.96 percent.

5 seconds duration were recorded, arranged as 5 trials each with both devices in the left hand, both devices in the right hand, one ball in each hand, while standing and while sitting. For each trial, the balls were picked up from a table and put down afterwards, so that the relative orientation of the balls changed from trial to trial.

The collection of the second data set involved pairs of subjects. The subjects were asked to shake one of the balls each and to attempt to create the same movement. To incentivize the participating teams, the data collection was organized as a competition with a small price for scoring the highest similarity in shaking movements. For each team, 20 trials of approximately 15 seconds duration were recorded: a longer period than in the collection of data from individuals to give the participants time to get synchronized. The 20 trials were organized as 5 trials each in which both subjects used their left hand, both their right hand, one left and one right, and vice versa. For all trials, the participants were standing and within close proximity (1 meter) of each other. After each trial, the teams were given immediate feedback on the similarity of their movements, so that they could attempt to adapt their shaking patterns with the aim to achieve a higher score.

7.2 Results

We recruited 51 participants locally in our University, 19 female and 32 male, aged between 20 and 58, including students, researchers, office staff, and nonoffice workers. All 51 subjects participated in the collection of the first data set, resulting in 1,530 samples of shaking by an individual subject. Two-thirds of the samples resulted from shaking together in one hand (either left, or right) and were classified as "positive" because authentication should succeed for this condition. The remaining third resulted from shaking of devices in one hand each and were classified as "neutral" because authentication could succeed (as the movement was performed by the same person) but would not necessarily be expected to succeed. Of the subjects, 24 participated also in the second data collection, in 12 teams of two. Of the teams, 8 performed the complete trial and the other 4 a smaller number of trials, with a total of 177 data sets collected as "negative" samples for the evaluation of our classification algorithms.

Figs. 5 and 6 show the performance of our algorithms in terms of false positive and false negative rates, depending on thresholds chosen for coherence and feature matching. The main result, for our purposes, is that both algorithms can be configured to robustly classify shaking events only as *shaking together* if the devices have indeed been shaken by the same person (i.e., thresholds can be set to robustly exclude false positives). This is an important result, as it confirms the utility, in principle, of shaking for secure authentication.

However, the two algorithms behave very differently. The first algorithm evaluates coherence of shaking patterns as a scalar measure of similarity (Fig. 5). As expected, the higher the threshold, the lower the false positive rate but the higher the false negative rate. The optimum for our purposes is at a threshold of $C_{xy} = 0.72$, where the false positive rate drops to zero, while the false negative rate is still low at 10.24 percent. Applying this threshold in the ShaVe method, all illegitimate authentication attempts are rejected, while almost 90 percent of the legitimate attempts succeed.

The second algorithm evaluates similarity of shaking movements in terms of the proportion of matching feature vectors (Fig. 6). The false positive rate produced by this algorithm drops with increasing threshold, but the false negative rate is nearly constant, below 12 percent. Applied in the ShaCK authentication method, this means that almost 90 percent of legitimate attempts produce a match for every candidate feature vector, and that thresholding at a lower proportion of matching features does not lead to any improvement in terms of accepting valid attempts. The step profile of the false positive rate is explained by the use of $B = 6$ quantization levels in the exchanged feature vectors. Above a threshold of 0.88 for proportion of matching feature vectors, the algorithm rejects any illegitimate authentication attempt. However, as the false negative rate does not increase with a higher threshold, the ShaCK method can be configured to require a 100 percent matching rate as authentication condition.

The analysis of the coherence of separately recorded data streams requires, in contrast to candidate feature matching, active segments of fixed duration. The results presented above are based on our default approach of taking the first 3 seconds of any simultaneous shaking event. However, if a shaking event has a longer duration, then it is also possible

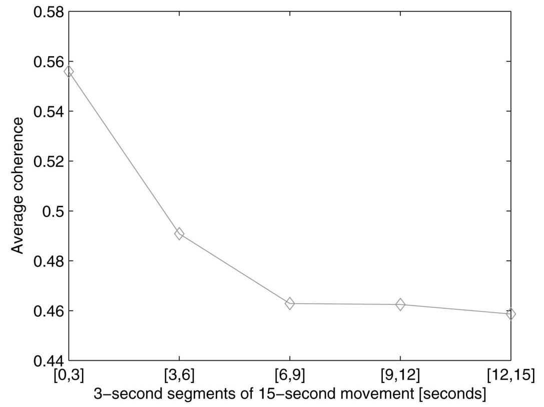


Fig. 7. The average coherence of consecutive three-second segments extracted from *independent shaking* decreases significantly over the first two segments.

to select active segments for coherence analysis that are offset from the start of the event. Fig. 7 shows an analysis based on the "negative" data set, with average coherence values plotted for active segments that represent the first 3 seconds, the second 3 seconds, and so on, of *independent shaking* by two subjects. The graph shows that the subjects tend to loose synchronization the longer the common movement needs to be sustained. This result suggests that the separability of positive and negative conditions can be further improved if common movement is evaluated only after it has already been sustained for a few seconds; however this would be at the expense of requiring users to shake devices longer in order to achieve authentication.

The parameters of our algorithms have been derived from an analysis of our data sets. For the coherence algorithm, we adopted the parameter set that generated the largest difference in coherence values between positive samples and negative samples. For the candidate feature matching algorithm, a parameter set was chosen that minimized $4e_P + e_N$, with e_P the false positive rate and e_N the false negative rate. Figs. 8 and 9 provide insight into the influence of individual parameters on algorithm performance. For the coherence algorithm, the sampling rate has a significant influence on the difference in coherence measured for our two conditions, with the "gap" widening with higher sampling rates but no further improvement from 256 to 512 Hz (Fig. 8a). In contrast, the choice of threshold used for detection of the onset of a shaking event (i.e., the start of an active segment) shows practically no impact on separability of our two conditions (Fig. 8b). For the cutoff frequency used in the coherence metric, our analysis confirms the intuitive correlation of better separation with higher values but with decreasing significance from around 20 Hz (Fig. 8c). For the window size in the coherence metric, best results were observed for values corresponding to the sampling rate (Fig. 8d). Overall, the data show that the coherence algorithm is robust against parameter variation and provides separability of *shaking together* and *shaking independently* irrespective of particular parameter settings.

The most significant parameters of the candidate feature matching algorithm are the number of candidates and the number of quantization levels used in the feature vector. For the number of candidates, it is expected that the matching rate increases with higher values, however, the increase is not significant for more than four candidates (Fig. 9a). The choice of quantization levels is particularly critical for separability of our two conditions in terms of matching rates, with best results observed for $B = 6$ levels (Fig. 9b). The sampling rate and cutoff frequency, on the other hand, show limited influence on separability of our two conditions (Figs. 9c and 9d). The data show that also this algorithm is robust against parameter variation for correct classification. However, as the algorithm is based on discrete matching of features, the classification performance is significantly more dependent on a "good" parameter combination than is the case for the coherence algorithm.

8 System Evaluation

The first part of our evaluation has shown that the algorithms used in our authentication methods can be used to robustly differentiate whether devices have been shaken together by an individual or shaken separately. In this section, we evaluate our methods for authentication by shaking on system level, with respect to three concerns: the usability of the approach, the feasibility of its implementation in mobile devices, and the security of the two concrete authentication methods, ShaVe and ShaCK.

8.1 User Study

8.1.1 Design and Procedure

A user study was designed to investigate how intuitive the proposed method of shaking is for device pairing. The test hypothesis was that users are able to learn the method with minimal guidance and effort, and that they can apply the

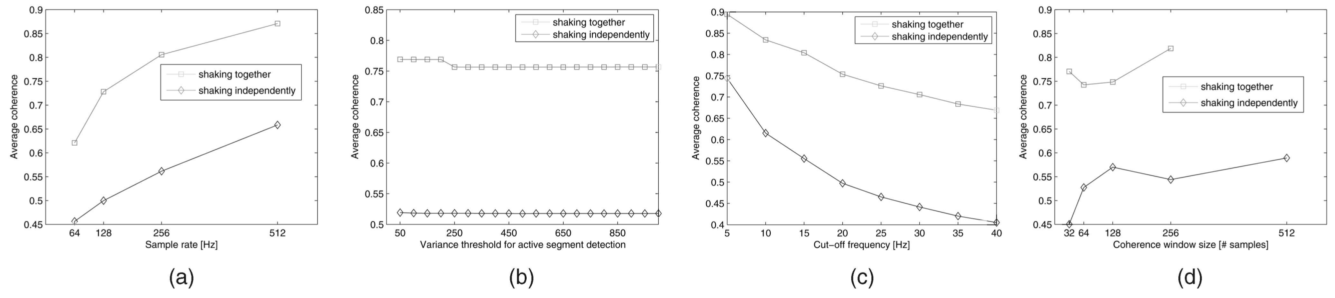
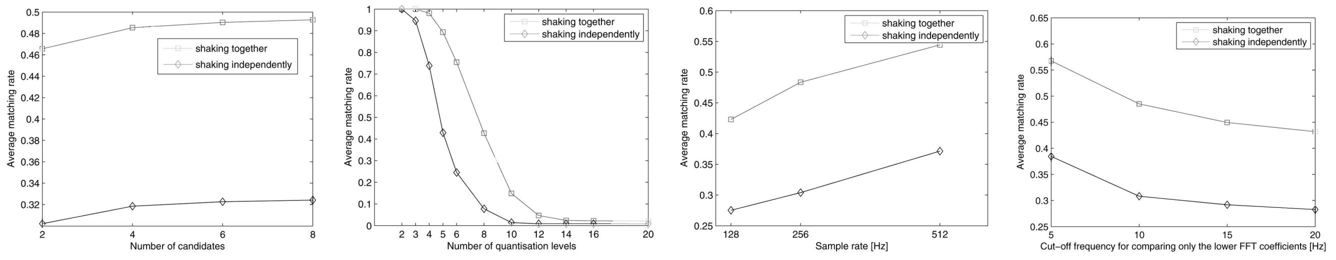


Fig. 8. Analysis of the influence of selected parameters on the performance of the coherence algorithm: (a) sampling rate, (b) signal variance threshold for segmentation, (c) cutoff frequency in the coherence metric, and (d) window size for computation of coherence.



(a) (b) (c) (d)

Fig. 9. Analysis of parameters used in the candidate feature matching algorithm: (a) number of candidate feature vectors, (b) number of quantization levels used in feature vectors, (c) sampling rate, and (d) cutoff frequency.

method reliably after they have learnt it. As apparatus for the study, the ping-pong balls previously used for data collection were now connected to a "live" system for online analysis of the acceleration time series (Fig. 10). The system used both coherence and candidate feature matching in parallel for the analysis. A simple graphical user interface (GUI) showed the status of both devices (active/inactive) and the values for coherence and feature matching rate, highlighted with a green background for values above the threshold and red background for values below, to indicate authentication success or failure. The user interface was updated whenever the algorithms produced a new results, including when users continued to shake devices for longer than a single trial.

The procedure for the user sessions was to very briefly explain the experimental setup and to provide users with a short list of hints on how to achieve successful pairing with the shaking method: to align the devices along the movement axis, to shake quickly and vigorously, and to keep the wrist stiff and the elbow steady. The users were not given a practical demonstration. They were then asked to try the method with the aim to achieve successful pairing, and keep trying in case they did not succeed immediately. After the first successful authentication (simultaneously with both methods, coherence and feature matching), participants were asked to reproduce this for at least two more times. When participants did not achieve successful authentication after 10 attempts (about 1 minute), they were shown once how the experimenter performs the method for successful authentication, and following the demonstration asked to try again.

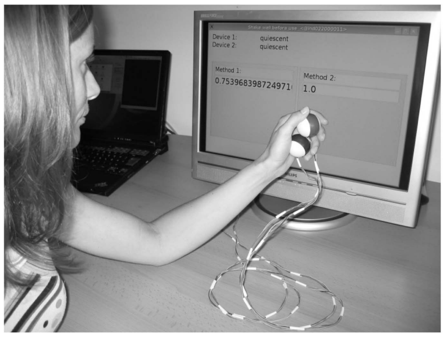


Fig. 10. Experimental setup for the user study: sensors embedded in ping-pong balls connected to a system that provided real-time feedback on authentication success or failure.

8.1.2 Results

A total of 30 participants (12 females, 18 males), aged 24 to over 50, took part in the study, recruited from staff, students, and visitors in our University. Of the participants, 70 percent had taken part in the previous data collection, and therefore, had already seen the sensing devices before.

The results of the user study are summarized in Table 1. As shown, 8 of the participants (26.6 percent) were able to immediately and reproducibly achieve successful authentication for both methods starting with their first try. A further 10 participants (33.3 percent) acquired the method successfully by trial and error, 8 within five attempts (about half a minute) and the other two within 10 attempts (about one minute). Of the remaining participants, seven (23.3 percent) were able to reproducibly achieve authentication after they were given a demonstration of a successful attempt by the experimenter, within at most three further attempts following the demo. The remaining five participants (16.7 percent) only achieved hits with either of the two methods (two only with ShaVe, three only with ShaCK), but did not succeed in reproducible authentication with both methods.

The main result of the study is that most users learn to use the shaking method for secure pairing with minimal preparation: a brief phase of trial and error, or a single *how-to* demonstration. This confirms that the method is intuitive and can be mastered by users without an explicit training phase. However, the study also shows that some users found it difficult to use the method successfully. We observed that unsuccessful users were inclined to shake carefully (i.e., with limited vigor) and/or with a strong rotation element in their movement, which are both detrimental to detection of *shaking together*.

8.2 Feasibility of Embedded Implementation

In order to demonstrate the feasibility of our approach to authentication on mobile devices, we have implemented the ShaVe method on the Nokia 5500, a state-of-the-art mobile

TABLE 1
Preparation Required by Users to Learn the Shaking Method

Preparation	*Number of users (%)*	Accumulated
No preparation	8 (26.6%)	
Trial and error (self-training)	10 (33.3%)	18 (60%)
Demonstration by instructor	7 (23.3%)	25 (83.3%)
Successful only with ShaVe	2 (6.6%)	
Successful only with ShaCK	3 (10%)	5 (16.6%)

Fig. 11. The ShaVe method was fully implemented on off-the-shelf Nokia 5500 mobile phones to demonstrate feasibility on mobile devices.

phone (Fig. 11). The ShaCK method would offer more dynamic interactions but requires wireless broadcast capabilities for direct implementation, which Bluetooth lacks. The Nokia 5500 series has a built-in 3D accelerometer that facilitated implementation of our method without any extension or modification of the mobile phone hardware. The implementation is based on a background Symbian application that is started automatically and uses the Nokia Sensor API to access the accelerometer data at its preset sampling rate of about 30 Hz. This sampling rate is significantly lower than the sampling rate that had been identified as best-performing in the experiments reported above. At this lower rate, *shaking together* and *shaking independently* can still be robustly distinguished at a lower coherence threshold, however, the false negative rate is then expected to be higher than in our experiments (i.e., a higher rate of rejecting valid attempts).

The core of our implementation is a Java MIDlet that connects via a local TCP socket to the Symbian background process. The MIDlet contains the code for processing of the acceleration data and uses Bluetooth RFCOMM communication for implementation of the key verification protocol. The use of Bluetooth as a wireless communication channel posed challenges, as Bluetooth lacks broadcast or multicast support, and because of the latency of inquiry and service discovery. We addressed this by performing the first phase of ShaVe, i.e., the Diffie-Hellman key agreement, opportunistically. Whenever a compatible device is found by the regular background inquiry process, an unauthenticated secret shared key is established with it. The second phase, key verification, is started as soon as a valid active segment has been detected and recorded. The acceleration data is encrypted and securely exchanged with all devices for which a shared key was generated in the first phase. Although the messages are distributed to potentially multiple devices, only the pair that has been shaken together will authenticate their secret shared key.

Cryptographic operations and signal processing obviously bear a cost; as an indicator for the performance of the embedded implementation on off-the-shelf mobile phones, the Diffie-Hellman key agreement phase takes about 6 seconds, with 4 seconds for computation and 2 seconds for transfer of key material via Bluetooth. The key verification phase is significantly faster with practically no delay for AES encryption, and about 1 second for Bluetooth transmission of encrypted active segments. All cryptographic primitives rely on integer operations and can, in practice, be executed on arbitrary CPUs. On modern mobile phones with CPU clock rates of 100 MHz and above, asymmetric cryptography (including Diffie-Hellman as used in ShaVe) is feasible without specialized hardware support.

A significant cost is associated with sensor data processing as this requires floating point operations. These are costly on embedded CPUs without hardware floating point units, and for our demonstrator, we therefore converted the algorithms to deal with integer operations only. FFT and coherence however are computed with floating point numbers and double accuracy, which takes about 0.3 seconds in our implementation on the Nokia 5500. On dedicated DSPs or processors with signal processing support, FFT computation cost would be lower and feasible on more resource-limited devices.

A further cost factor is wireless transmission. ShaVe always requires eight messages for a complete protocol run (two for key agreement, four for verification via interlock, and two for the final acknowledgment). The size of interlock verification messages depends on how long devices were shaken, and therefore, scales linearly with the entropy (and in this case, security against online attacks). For ShaCK, messages sizes are constant, but the number of candidate key part messages scales with the shaking duration (and thus with the security against online and offline attacks). At a minimum, ShaCK will transmit six messages (which are shorter than those in ShaVe), but with no upper limit.

For overall cost analysis, it is important to note that shaking for pairing will occur infrequently and only as short burst of activity. In the normal lifecycle of embedded devices such as mobile phones and accessories, impact on battery life is therefore expected to be negligible.

8.3 Security Analysis

In the following discussion, we assume an adversarial model similar to the one used by Vaudeney [36]. That is, the adversary has full control over the wireless channel, including reading, modifying (data or address of) a message, replaying, preventing delivery, or injecting completely new messages. Additionally, the adversary may start new protocol instances (by injecting appropriate authentication request messages) with multiple legitimate devices in parallel—especially with two devices being intentionally shaken together by the user. Although the adversary may also monitor all legitimate devices, we assume that they can neither control the acceleration experienced locally by these devices nor perfectly estimate it.

8.3.1 *Threat Scenarios*

The primary threat for authentication is an active MITM attack. For ShaVe, MITM attacks on the wireless channel are prevented by the key verification phase binding the agreed session keys to sensor time series that are verifiable by both devices. An attacker might succeed in completing the Interlock exchange of sensor data (by sending a junk message after receipt of the first part of the data) but the receiving device would detect the attack and signal this to the user. For ShaCK, session keys are not explicitly negotiated, but are computed internally based on matching received key material with local sensor time series. These keys are therefore resistant to classical attacks on key agreement.

As our authentication methods are based on shared movement, we need to consider three particular threat scenarios for the detection of such shared movement. First, devices may be accidentally paired by unintentional joint movement. This threat is prevented in ShaVe and ShaCK by minimum thresholds on both shaking time and vigor, so that unintentional movement is highly unlikely to generate acceleration patterns that are detected as active segments.

Second, an adversary may try to physically copy the movements of legitimate users, for example, while they are attempting to pair devices. This threat is addressed by robust classification of *shaking together* versus *shaking independently* as analyzed above in Section 7. Our analysis shows that our decision algorithms can be parametrized to prevent false positives, even for the case of two users actively collaborating to "fool" the system by attempting to perform the same movement. We conclude that an adversary who might try to perform the same shaking movement as the user (without their cooperation) would not succeed in producing a sufficiently similar sensor data.

Third, an adversary may try to estimate the accelerations the attacked device is experiencing, for example, based on video analysis, and fabricate their own time series to be sufficiently similar for successful authentication by the other device. The basic assumption of ShaVe and ShaCK is that there is sensor data that the legitimate devices can capture with better accuracy than any adversary. The devices measure their acceleration as a physical phenomenon that is *local* and which an adversary cannot directly access or reasonably influence. To prevent the threat of "guesstimating" acceleration time series, the most important point is to achieve high entropy with regards to a possible adversary's knowledge. A key can only remain secret if the adversary is sufficiently uncertain about it. Interactive cryptographic protocols can "stretch" this entropy when creating a key, but they still require sufficient uncertainty so that an adversary can not estimate the whole sensor time series during a single authentication protocol run. The main difference between ShaVe and ShaCK in this respect is their resistance against offline guessing attacks.

8.3.2 Resistance of ShaVe against Guessing Attacks

ShaVe uses two phases, *key agreement* and *key verification*, and thus offers a potential adversary only a *single, one-off chance*—to estimate the respective sensor time series closely enough and online during the protocol run—to undetectedly mount an MITM attack. When the messages sent in the Interlock* protocol, that is, the active segments extracted from acceleration time series, each have an entropy of e bits from the adversary's point of view, this leaves a single 2^{-e} chance of remaining undetected. We can currently not quantify the entropy of these messages, but a guessing attack seems prohibitively unlikely in practice. Note that, even with e less than 128 bits, the created session keys will still offer the full security level (assuming DH to be secure).

8.3.3 Resistance of ShaCK against Guessing Attacks

For ShaCK, the security level of this CKP-based protocol is directly limited by the entropy of all collected feature vectors and is susceptible to offline brute-force attacks (cf. the security analysis in [35]). This is due to relying solely on symmetric cryptographic primitives. By optionally using any standard password-based key agreement protocol such as the MANA family of protocols [37] with the candidate key K as common, secret input, ShaCK could also be hardened against such offline attacks. However, this requires the introduction of asymmetric cryptography with the associated computational cost. We therefore analyze the security of ShaCK without this optional protocol run and note that, when offline security against brute-force attacks is required, ShaVe or ShaCK with an added asymmetric key agreement protocol should be used.

The problem arises when parts that are extracted from sensor time series only have a small entropy from an adversary's point of view. In this situation, even when reversing the hash function is impossible, the adversary could just generate lookup tables of all possible time series parts and compare their hashes with the received candidate key parts. This is slightly mitigated by the use of salting, but only makes the attack more computationally expensive, and not less likely to be successful. An adversary only needs to keep a small amount of possibly matching key parts in a history to try and create keys that match the transmitted messages, in much the same way that is also used in the legitimate protocol run. For this reason, it is better to transmit fewer but larger candidate key parts to construct a key. When the sensor time series parts that can be extracted naturally using domain specific knowledge only have a small entropy, then multiple such parts should be buffered and bundled into one candidate key part. This does not reduce the overall entropy collected from as much sensor data as possible, but makes attacks on the transmitted parts harder. Guessing a candidate key part and verifying that it matches its received hash value has an average complexity of $O(2^{e-1})$ when the feature vector has e bits of entropy from the adversary's point of view. Thus, two concatenated feature vectors would need $O(2^{2e-1})$ steps to guess.

For an estimate of the entropy e of quantized feature vectors used for ShaCK, we further analyzed the collected shaking data. Using the parameters found with the first two data sets but 256 Hz instead of 512 Hz sample rate, quantized FFT coefficient vectors were computed over all 1,530 samples. This parameter combination generates feature vectors of 21 discrete values from 0 to 5. Each subject generated, on average, 526.86 different feature vectors, with a minimum of 140 and a maximum of 1,037. Aggregated over all subjects, there were 5,595 different vectors for the left hands, 4,883 for the right, and 7,988 and 7,770 for devices 1 and 2, respectively. Overall, 12,220 different feature vectors were generated during the first experiment, corresponding to an entropy of $ld12,220 = 13.58$ bits per feature vector. If an adversary is assumed to know which device, person, and hand are involved in a protocol run, this entropy decreases to around 7 (for the subject generating only 140 different feature vectors, pessimistically reduced by the assumption of which device and hand was used) to 10 bits (for the single subject with 1,037 different vectors). Overlapping feature vectors will have even less entropy, but we currently assume ShaCK to generate 7 bits entropy per second, and thus, to achieve a security level of 128 bits after around 20 seconds. Powerful motion-based estimation methods, for example, based on video analysis, may give adversaries additional information, and thus, reduce their entropy further. However, such an attack seems unlikely in practice. Users or applications may choose lower (or higher) security levels simply by deciding how long devices should be shaken.

A potential concern is that secure hash functions, the cornerstone of the CKP design, have been subjected to considerably less theoretical analysis than other cryptographic primitives. ShaVe also uses hash functions but only to derive symmetric session keys from Diffie-Hellman keys, and thus, seems less vulnerable against potential future attacks on hashes than ShaCK.

9 Conclusion

In this paper, we have introduced authentication by shaking as a novel concept for pairing of mobile devices. We have developed and validated algorithms for robust classification of shaking movement, and contributed protocols that interweave sensor data analysis with cryptographic techniques for authentication. The key findings of this research are that authentication by shaking can provide strong security for device pairing; that the technique is intuitive and easy to perform by users; and that it is feasible to implement the approach on mobile embedded platforms with minimal input/output capacity.

The discussed research provides insight into alternative designs for authentication by shaking. The two concrete methods developed in this work are based on the same sensor data as input but combine data analysis and cryptographic protocols in very different ways. ShaVe is based on consecutive steps of speculative key agreement, exchange of sensor data, and finally, independent verification of the agreed key. This conservative method results in strong security but is more rigid with consecutive phases, and predefined length of sensor time series required for comparison. Note that the independent and noninteractive comparison of "raw" sensor data allow devices to apply different methods for key verification, and thus, affords a significant advantage for interoperability and different security levels. ShaCK, in contrast, follows a pipeline design, with feature extraction, exchange of candidate features, and key construction stages. The result is a very interactive technique where sensor data is continually processed, and where a shared key is gradually built up the longer a pair of devices are shaken together. If strong security is desired, ShaCK requires shaking over a longer duration than ShaVe. However, ShaCK provides the flexibility to compromise key strength for faster authentication and can be more efficient than ShaVe if security comparable to four-digit PIN codes is sufficient. A specific advantage of ShaCK is the immediacy of user feedback, configurable to signal success as soon as a target level of security has been reached (whereas ShaVe involves inherent latency as data is analyzed over time windows of several seconds).

The security of the proposed method depends on the combination of algorithms for data analysis and cryptographic protocols. However, the design is modular with a clear separation of data capture, data analysis, and protocols. Performance improvements can be achieved, for instance, in data preprocessing (e.g., to obtain higher entropy or limit off-center rotational effects) without modification of protocols, and only limited impact on classification algorithms. The data analysis algorithms are developed specifically for acceleration time series, but the cryptographic protocols could also be used with other sensors and data analysis methods.

Based on the reported results, we would expect improvements from spatial alignment of sensor data in three dimensions. Presently, only the magnitude of acceleration values is captured, and the direction of the vectors could contribute significantly to the overall entropy and the security of our authentication method. This can be approached with modeling of movement in 3D and estimation of relative spatial alignment between the devices, to reconcile the local and remote sensor time series. For practical deployment, it may also be relevant to support negotiation of parameters for data analysis and for required security.

Authentication by shaking is well suited to pairing of small-size devices that can be held together in one hand. Secure pairing is specifically relevant for personal devices, such as health monitors worn on the body, or smart cards and pass keys. The hardware requirements for the proposed technique are minimal; a single accelerometer is required to capture movement, and an LED, beeper or vibrator, would suffice for user feedback. Many mobile devices already incorporate the required hardware, as demonstrated for mobile phones, and it is reasonable to expect wider deployment of movement sensors in devices for a variety of purposes (e.g., device monitoring, gesture detection, context-awareness). A potential limitation for deployment on very small devices are the computational resources required for advanced data analysis. The impact on battery lifetime of embedded devices, however, is considered minimal, as shaking-for-pairing involves only short bursts of sensor, CPU, and radio usage.

This work has explored use of movement as evidence for securing spontaneous device association. A challenge in future work is to advance toward more principled understanding of how sensing of physical phenomena can contribute to practical security in mobile and ubiquitous computing.

The data sets and source code of prototype implementations are available as open source at http://www.openuat.org.

Appendix A
Protocol Specifications

Figs. 12 and 13 show the detailed specifications of the cryptographic protocols for ShaVe and ShaCK, respectively. For the formal descriptions of the protocols, the following notations are used: $c = E(K, m)$ describes the encryption of plain text m under key K with a symmetric cipher, $m = D(K, c)$ the corresponding decryption, $H(m)$ describes the hashing of message m with some secure hash, and $m|n$ the concatenation of strings m and n. The notation $M[a:b]$ is used to describe the substring of a message M starting at bit a and ending at bit b. The symbol $\oplus$ describes bitwise

A	Message	**B**
	Choose DH key	
$x \in \{1 \ldots q-1\}$	$X := g^x$	$y \in \{1 \ldots q-1\}$
$K_a^{Sess} :=$	$\longrightarrow$	$K_b^{Sess} :=$
$H(Y^x)$	$Y := g^y$	$H(X^y)$
$K_a^{Auth} :=$	$\longleftarrow$	$K_b^{Auth} :=$
$H(Y^x \vert C)$		$H(X^y \vert C)$
	Split and encrypt signal	
$(i = 0 : n-1, \quad j = 1 : n)$:	$A_1 = c_0[0:63] \vert \ldots \vert c_n[0:63]$	$(i = 0 : m-1, \quad j = 1 : m)$:
$a_i := a[128i : 128i+127]$	$\longrightarrow$	$b_i := b[128i : 128i+127]$
$c_0 \in \{0, ..., 2^{128}-1\}$	$B_1 = d_0[0:63] \vert \ldots \vert d_m[0:63]$	$d_0 \in \{0, ..., 2^{128}-1\}$
$c_j := E(K_a^{Auth}, c_{j-1} \oplus a_{j-1})$	$\longleftarrow$	$d_j := E(K_b^{Auth}, d_{j-1} \oplus b_{j-1})$
	Interlock and reassemable	
$(i = 0 : m-1)$:	$A_2 = c_0[64:127] \vert \ldots \vert c_n[64:127]$	$(i = 0 : n-1)$:
$\tilde{d}_i := \tilde{B}_1[128i : 128i+63] \vert$	$\longrightarrow$	$\tilde{c}_i := \tilde{A}_1[128i : 128i+63] \vert$
$\tilde{B}_2[128i : 128i+63]$	$B_2 = d_0[64:127] \vert \ldots \vert d_n[64:127]$	$\tilde{A}_2[128i : 128i+63]$
$\tilde{b}_i := D(K_a^{Auth}, \tilde{d}_{i+1}) \oplus \tilde{d}_i$	$\longleftarrow$	$\tilde{a}_i := D(K_b^{Auth}, \tilde{c}_{i+1}) \oplus \tilde{c}_i$
$\tilde{b} := \tilde{b}_0 \vert \ldots \vert \tilde{b}_m$		$\tilde{a} := \tilde{a}_0 \vert \ldots \vert \tilde{a}_n$
	Compare	
if $a \sim \tilde{b}$		if $b \sim \tilde{a}$
then $K := K_a^{Sess}$		then $K := K_b^{Sess}$
else $K := null$		else $K := null$

Fig. 12. ShaVe protocol: Diffie-Hellman key agreement followed by exchange of the complete time series via Interlock*.

A	Message	**B**
$s_a^{r_a} \in \{0, \ldots, 2^{128}-1\}$	$C_a^{r_a} :=< r_a, s_a^{r_a}, h_a^1, h_a^2, \ldots >$	$s_b^{r_b} \in \{0, \ldots, 2^{128}-1\}$
For each vector v_a^i:	$\longrightarrow$	**For each vector** v_b^j:
$h_a^i := H(s_a \vert v_a^i)$	$C_b^{r_b} :=< r_b, s_b^{r_b}, h_b^1, h_b^2, \ldots >$	$h_b^j := H(s_b \vert v_b^j)$
add $< r_a, v_a^i >$ to LH_a	$\longleftarrow$	add $< r_b, v_b^j >$ to LH_b
$r_a := r_a + 1$		$r_b := r_b + 1$
For each hash $\tilde{h}_b^j$:		**For each hash** $\tilde{h}_a^i$:
if $\exists < \hat{r}_a, \hat{v}_a^i > \in LH_a$		if $\exists < \hat{r}_b, \hat{v}_b^j > \in LH_b$
s.t. $\tilde{h}_b^j = H(\tilde{s}_b^{r_b} \vert \hat{v}_a^i)$		s.t. $\tilde{h}_a^i = H(\tilde{s}_a^{r_a} \vert \hat{v}_b^j)$
then add $< \hat{r}_a, \hat{v}_a^i >$ to $MC_{a,b}$		then add $< \hat{r}_b, \hat{v}_b^j >$ to $MC_{b,a}$
if $\vert MC_{a,b} \vert >= N_a$	$K_a :=< N_a, s_a^{r_a}, kh_a >$	if $\vert MC_{b,a} \vert >= N_b$
then $kh_a := H(s_a^{r_a} \vert last_{N_a}(MC_{a,b}))$	$\longrightarrow$	then $kh_b := H(s_b^{r_b} \vert last_{N_b}(MC_{b,a}))$
and $k_a := H(s_a^{r_a} \vert last_{N_a}(MC_{a,b}) \vert C)$	$K_b :=< N_b, s_b^{r_b}, kh_b >$	and $k_b := H(s_b^{r_b} \vert last_{N_b}(MC_{b,a}) \vert C)$
	$\longleftarrow$	
if $\exists \hat{c}_a \subseteq MC_{a,b}$	ACK kh_b	if $\exists \hat{c}_b \subseteq MC_{b,a}$
s.t. $H(\tilde{s}_b^{r_b} \vert \hat{c}_a) = \widetilde{kh}_b$	$\longrightarrow$	s.t. $H(\tilde{s}_a^{r_a} \vert \hat{c}_b) = \widetilde{kh}_a$
then $k_b' := H(\tilde{s}_b^{r_b} \vert \hat{c}_a \vert C)$	ACK kh_a	then $k_a' := H(\tilde{s}_a^{r_a} \vert \hat{c}_b \vert C)$
	$\longleftarrow$	
if k_a is not set $k := k_b'$		if k_b is not set $k := k_a'$
if k_b' is not set $k := k_a$		if k_a' is not set $k := k_b$
if $kh_a = \widetilde{kh}_b$		if $kh_b = \widetilde{kh}_a$
then $k := k_a$		then $k := k_b$
else $k := k_a \oplus k_b'$		else $k := k_b \oplus k_a'$

Fig. 13. ShaCK protocol: candidate key protocol (CKP) for directly creating a secret key from common feature vector hashes.

XOR and $|S|$ the number of elements in a set S. If a message M is transmitted over an insecure channel, the received message is denoted by $\widetilde{M}$ to point out that it may have been modified in transit, by noise or attack. Subscripts denote the different sides (a or b for an authentication between A and B) while superscripts denote specific vectors in a set of vectors. The syntax $\widehat{x}$ denotes the (open) result of a search for matches in a set. When a hash is computed from a set of vectors, this means the concatenation of all vectors in some predefined order, typically by their round number. C refers to some publicly known constant. AES is used as as a block cipher for E and D and $SHA_{DBL} - 256$ as a secure hash for H, which is a double execution of the standard $SHA - 256$ message digest to safeguard against length extension and partial-message collision attacks and is defined as $SHA_{DBL} - 256(m) = SHA - 256((SHA - 256(m))|m)$.

References

[1] F. Stajano and R. Anderson, "The Resurrecting Duckling: Security Issues for Ad-Hoc Wireless Networks," *Proc. Seventh Int'l Workshop Security Protocols,* pp. 172-194, Apr. 1999.

[2] T. Kindberg and K. Zhang, "Validating and Securing Spontaneous Associations between Wireless Devices," *Proc. Information Security Conf. (ISC '03),* pp. 44-53, Oct. 2003.

[3] C. Gehrmann, C.J. Mitchell, and K. Nyberg, "Manual Authentication for Wireless Devices," *RSA Cryptobytes,* vol. 7, no. 1, pp. 29-37, 2004.

[4] F. Stajano, "Security for Whom?: The Shifting Security Assumptions of Pervasive Computing," *Proc. Int'l Symp. System Synthesis (ISSS '02),* pp. 16-27, Nov. 2002.

[5] R. Want, T. Pering, G. Danneels, M. Kumar, M. Sundar, and J. Light, "The Personal Server: Changing the Way We Think about Ubiquitous Computing," *Proc. Int'l Conf. Ubiquitous Computing (UbiComp '02),* pp. 194-209, Sept. 2002.

[6] Bluetooth SIG, "Bluetooth Special Interest Group," Simple Pairing Whitepaper (Revision V10r00)," 2006.

[7] M. Čagalj, S. Čapkun, and J.-P. Hubaux, "Key Agreement in Peer-to-Peer Wireless Networks," *Proc. IEEE,* special issue on cryptography and security, vol. 94, pp. 467-478, 2006.

[8] A. Perrig and D. Song, "Hash Visualization: A New Technique to Improve Real-World Security," *Proc. Cryptographic Techniques and Electronic Commerce (CrypTEC '99),* pp. 131-138, 1999.

[9] N. Saxena, J.-E. Ekberg, K. Kostiainen, and N. Asokan, "Secure Device Pairing Based on a Visual Channel," Report 2006/050, Cryptology ePrint Archive, 2006.

[10] V. Roth, W. Polak, E. Rieffel, and T. Turner, "Simple and Effective Defense against Evil Twin Access Points," *Proc. ACM Conf. Wireless Network Security (WiSec '08),* pp. 220-235, Mar. 2008.

[11] M.T. Goodrich, M. Sirivianos, J. Solis, G. Tsudik, and E. Uzun, "Loud and Clear: Human Verifiable Authentication Based on Audio," *Proc. Int'l Conf. Distributed Computing Systems (ICDCS '06),* p. 10, July 2006.

[12] C. Soriente, G. Tsudik, and E. Uzun, "BEDA: Button-Enabled Device Pairing," *Proc. Int'l Workshop Security for Spontaneous Interaction (IWSSI 2007),* pp. 443-449, Sept. 2007.

[13] S.N. Patel, J.S. Pierce, and G.D. Abowd, "A Gesture-Based Authentication Scheme for Untrusted Public Terminals," *Proc. ACM Symp. User Interface Software and Technology (UIST '04),* pp. 157-160, Oct. 2004.

[14] D. Balfanz, D.K. Smetters, P. Stewart, and H.C. Wong, "Talking to Strangers: Authentication in Ad-Hoc Wireless Networks," *Proc. Network and Distributed Systems Security Symp. (NDSS '02),* Feb. 2002.

[15] D. Balfanz, G. Durfee, R.E. Grinter, D.K. Smetters, and P. Stewart, "Network-in-a-Box: How to Set up a Secure Wireless Network in under a Minute," *Proc. 13th USENIX Security Symp.* pp. 207-222, Aug. 2004.

[16] T. Kindberg and K. Zhang, "Secure Spontaneous Device Association," *Proc. Int'l Conf. Ubiquitous Computing (UbiComp '03),* pp. 124-131, Oct. 2003.

[17] R. Mayrhofer and M. Welch, "A Human-Verifiable Authentication Protocol Using Visible Laser Light," *Proc. Int'l Conf. Availability, Reliability and Security (ARES '07),* pp. 1143-1147, Apr. 2007.

[18] R. Mayrhofer, H. Gellersen, and M. Hazas, "Security by Spatial Reference: Using Relative Positioning to Authenticate Devices for Spontaneous Interaction," *Proc. Int'l Conf. Ubiquitous Computing (Ubicomp '07),* pp. 199-216, Sept. 2007.

[19] A. Varshavsky, A. Scannell, A. LaMarca, and E. de Lara, "Amigo: Proximity-Based Authentication of Mobile Devices," *Proc. Int'l Conf. Ubiquitous Computing (UbiComp '07),* pp. 253-270, Sept. 2007.

[20] J. Rekimoto, Y. Ayatsuka, and M. Kohno, "SyncTap: An Interaction Technique for Mobile Networking," *Proc. Int'l Conf. Mobile Human-Computer Interaction (MOBILE HCI '03),* pp. 104-115, Sept. 2003.

[21] J.M. McCune, A. Perrig, and M.K. Reiter, "Seeing-Is-Believing: Using Camera Phones for Human-Verifiable Authentication," *Proc. IEEE Symp. Security and Privacy,* pp. 110-124, 2005.

[22] L.E. Holmquist, F. Mattern, B. Schiele, P.A., M. Beigl, and H.-W. Gellersen, "Smart-Its Friends: A Technique for Users to Easily Establish Connections between Smart Artefacts," *Proc. Int'l Conf. Ubiquitous Computing (UbiComp '01),* pp. 116-122, Sept. 2001.

[23] K. Hinckley, "Synchronous Gestures for Multiple Persons and Computers," *Proc. ACM Symp. User Interface Software and Technology (UIST '03),* pp. 149-158. Nov. 2003.

[24] J. Lester, B. Hannaford, and G. Borriello, "Are You with Me?—Using Accelerometers to Determine If Two Devices Are Carried by the Same Person," *Proc. Pervasive Computing Int'l Conf. (PERVASIVE '04),* pp. 33-50, 2004.

[25] R. Marin-Perianu, M. Marin-Perianu, P. Havinga, and H. Scholten, "Movement-Based Group Awareness with Wireless Sensor Networks," *Proc. Pervasive Computing Int'l Conf. (PERVASIVE '07),* pp. 298-315, 2007.

[26] K. Fujinami and S. Pirttikangas, "A Study on a Correlation Coefficient to Associate an Object with Its User," *Proc. AAAI Spring Symp. Intelligent Environments (IE '07),* pp. 288-295, Sept. 2007.

[27] R. Mayrhofer and H. Gellersen, "Shake Well Before Use: Authentication Based on Accelerometer Data," *Proc. Pervasive Computing Int'l Conf. (PERVASIVE '07),* pp. 144-161, May 2007.

[28] D. Kirovski, M. Sinclair, and D. Wilson, "The Martini Synch," Technical Report MSR-TR-2007-123, Microsoft Research, Sept. 2007.

[29] D. Bichler, G. Stromberg, M. Huemer, and M. Löw, "Key Generation Based on Acceleration Data of Shaking Processes," *Proc. Int'l Conf. Ubiquitous Computing (UbiComp '07),* pp. 304-317, 2007.

[30] C. Castelluccia and P. Mutaf, "Shake Them Up! A Movement-Based Pairing Protocol for CPU-Constrained Devices," *Proc. Int'l Conf. Mobile Systems, Applications, and Services (MobiSys '05),* pp. 51-64, June 2005.

[31] L. Batina, N. Mentens, and I. Verbauwhede, "Side Channel Issues for Designing Secure Hardware Implementations," *Proc. IEEE Online Testing Symp. (IOLTS '05),* pp. 118-121, 2005.

[32] T. Huynh and B. Schiele, "Analyzing Features for Activity Recognition," *Proc. Joint Conf. Smart Objects and Ambient Intelligence: Innovative Context-Aware Services: Usages and Technologies (Soc-EUSAI '05),* pp. 159-163, Oct. 2005.

[33] W. Diffie and M.E. Hellman, "New Directions in Cryptography," *IEEE Trans. Information Theory,* vol. IT-22, no. 6, pp. 644-654, 1976.

[34] R.L. Rivest and A. Shamir, "How to Expose an Eavesdropper," *Comm. ACM,* vol. 27, no. 4, pp. 393-394, 1984.

[35] R. Mayrhofer, "The Candidate Key Protocol for Generating Secret Shared Keys from Similar Sensor Data Streams," *Proc. European Workshop Security and Privacy in Ad Hoc and Sensor Networks (ESAS '07),* pp. 1-15, July 2007.

[36] S. Vaudenay, "Secure Communications Over Insecure Channels Based on Short Authenticated Strings," *Proc. Ann. Int'l Cryptology Conf. (CRYPTO '05).* Aug. 2005.

[37] S. Laur and K. Nyberg, "Efficient Mutual Data Authentication Using Manually Authenticated Strings," *Proc. Int'l Conf. Cryptology and Network Security (CANS '06),* pp. 90-107, Dec. 2006.

Rene Mayrhofer received the Dipl-Ing (MSc) and Dr techn (PhD) degrees from Johannes Kepler University Linz, Austria, and subsequently received a Marie Curie Fellowship at Lancaster University, United Kingdom. He is currently a guest professor in mobile computing at the University of Vienna, Austria. His research interests include computer security, ubiquitous computing, and machine learning, which he brings together in his research on intuitive and unobtrusive techniques for securing spontaneous interaction.

Hans Gellersen received the MSc and PhD degrees in computer science from the University of Karlsruhe, Germany. He is currently a professor of interactive systems in the Department of Computing at Lancaster University, United Kingdom. His research interest is in ubiquitous computing, ranging from location and context sensing to user interface technologies.