Gait-Based Smart Pairing System for Personal Wearable Devices

Weitao Xu and Guohao Lan

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/intechopen.74195

Abstract

With the rapid development of embedded technology and mobile computing, we have seen a growing number of Internet of Things (IoT) devices on the market. As the number of wearable devices belonging to the same user increases rapidly, secure pairing between legitimate devices becomes an important research problem. In this chapter, we propose the first gait-based shared key generation system that assists two devices to generate a common secure key by exploiting the user's unique walking pattern. The system is based on the fact that sensors on different positions of the same user exhibit similar accelerometer signal when the user is walking. Therefore, the acceleration can be used as a shared secret information to generate a common key on different devices independently. Our experimental results show that the key generated by two independent devices on the same body is able to achieve 100% bit agreement rate. The proposed key generation protocol can establish a 128-bit key in 5 s (about 10 steps) with entropy varying from 0.93 to 1. We also find that the proposed scheme can run in real time on modern smartphone and require low system cost.

Keywords: wearable devices, authentication, key generation, system implementation, evaluation

1. Introduction

With recent advances in wireless sensor networks and embedded computing technologies, wearable and implantable devices such as smartphone, smartwatch and pacemaker have become increasingly popular and play significant roles in our daily lives. For users, it is of potentially great value to associate a personal device with another device in a spontaneous manner, i.e. without need for pre-configuration between two legitimate devices. Pairing devices can be used for the purpose of short-lived interactions, for example, file transfer and synchronization, or aimed at longer lived pairing, for example, pairing a smartphone with accessories.

The wireless nature of the communication between devices gives rise to problems of authentication and security [1]. **Figure 1** presents an example of on-body devices with potential adversaries. An adversary can listen to the communication between legitimate devices and eavesdrop private information. Traditional security mechanisms rely on cryptographic keys to support integrity and confidentiality services. Usually, two parties will first establish a common key, and then use the key to encrypt/decrypt subsequent communications between these two devices. In a dynamic mobile environment, mobile devices need to establish point-to-point association frequently. However, it is difficult to ensure the availability of a certificate authority or a key management center. Therefore, it is necessary to have alternative method for key distribution between mobile devices without resorting to a fixed infrastructure.

In current mobile systems, this is achieved by key exchange methods, which are either manual (e.g. typing in the key in a keypad) or exploit key-exchange algorithms. For the first case, a common mechanism for peer device authentication is personal identification number (PIN) code entry by the user into the involved devices [2]. Another example is that the user needs to make sure two devices show the same PIN code on the display when pairing devices using Bluetooth.

However, a primary requirement for human-involved authentication is ease of use and can interact with another device without much user intervention. Such methods offer a reasonably secure way of establishing a common key, but can only be acceptable for occasional use. The number of pairing between wearable devices can be expected to grow considerably as mobile

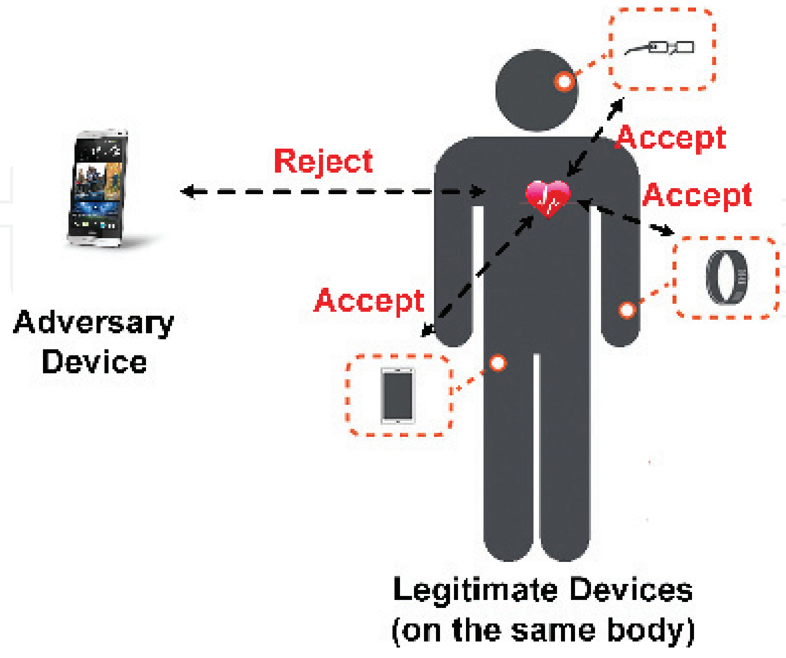

Figure 1. Legitimate device vs. adversary device.

devices are becoming increasingly pervasive. Therefore, the human-involved authentication method is undesirable when users seek to engage in fast and short-lived authentications frequently. To address the problem of pairing devices frequently, one can save the secure key on the device and use it next time. However, this method cannot ensure the security of the key because the stored key can be stolen. On the other hand, due to the small form factor and user interface (UI) of wearable devices, such method is not well suitable because wearable devices are not assumed to include screen and keyboard, e.g. pacemaker and health monitor. For the key-exchange algorithm, a common key exchange method is Diffie-Hellman (DH) protocol which is used to distribute a symmetric key between two parties [3]. However, the premise of DH protocol is that two devices to be paired together are legitimate devices. It cannot be used to distinguish adversary device with legitimate devices.

In this chapter, we propose and implement a shared secret key generation protocol for smart wearable devices based on gait. Gait refers to a person's manner of walking [4]. The intuition of the proposed key generation protocol is that devices on the same body experience similar signals when the user is walking. Therefore, the similar signals can be used to generate a shared secret key between legitimate devices. The proposed protocol provides an intuitive, unobtrusive method to pair wearable devices when they are on the same body. To the best of our knowledge, this is the first work to generate a key for wearable devices based on gait. The main contributions of this paper are two-fold:

1. Shared key generation protocol: We present a novel, lightweight key generation protocol for wearable devices based on gait signals. We experimentally demonstrate that the keys generated on two separate wearable devices on the same body can achieve a 100% bit agreement rate. The proposed key generation protocol is able to generate a 128-bit key with entropy varying from 0.93 to 1 by walking 5 s (≈10 steps).

2. Implementation: We implement the proposed key generation protocol on modern smartphone. We report system overhead such as processing time and power consumption, and demonstrate the feasibility of the proposed protocol for use in contemporary wearable devices.

The rest of the chapter is organized as follows. We first introduce system model in Section 2. Then we specify the shared key generation protocol in Section 3. We evaluate the performance of the proposed key generation protocol in Section 4 and implement the system in Section 5. Finally, Section 6 concludes the chapter.

2. System model

We envision the use of the proposed system primarily for pairing wearable and implantable devices such as smartphone, smartwatch and pacemaker. **Figure 2** illustrates a typical user model: a user wants to read private health information from pacemaker (Bob) through a smart watch (Alice). He walks several steps, and then Alice and Bob generate a shared secret key by exploiting the gait signals. The key is then used to encrypt/decrypt the messages between two parties.

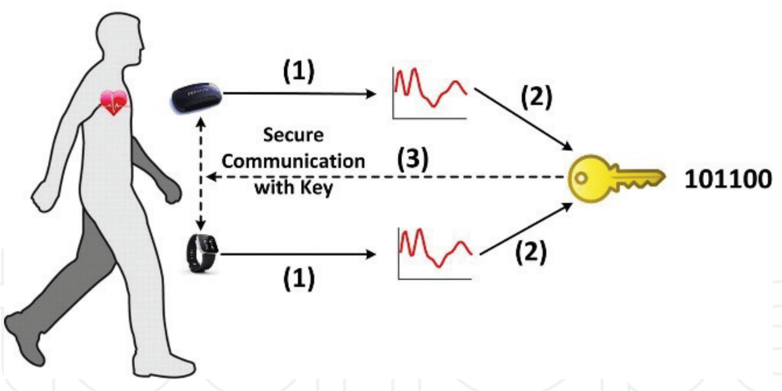

Figure 2. System model.

3. Design overview

In this section, we will provide an overview of the proposed scheme. **Figure 3** illustrates the flowchart of the key generation protocol. Suppose Alice (smartphone) wants to read data from Bob (pacemaker). Alice first broadcasts a *REQ* request to Bob. After receiving *REQ*, Bob replies a *ACK* message back to Alice. Then Alice and Bob start to collect data and follow the steps shown in **Figure 3** to generate a shared secret key. Finally, the key is used to secure the subsequent communication between Alice and Bob. The key component of the proposed scheme consists of the following two steps:

1. Signal processing. Signal processing consists of two steps: temporal alignment and spatial alignment. Temporal alignment is used to synchronize acceleration samples at Alice and Bob. Spatial alignment is used to transform the acceleration data to the same coordinate system to facilitate key generation.

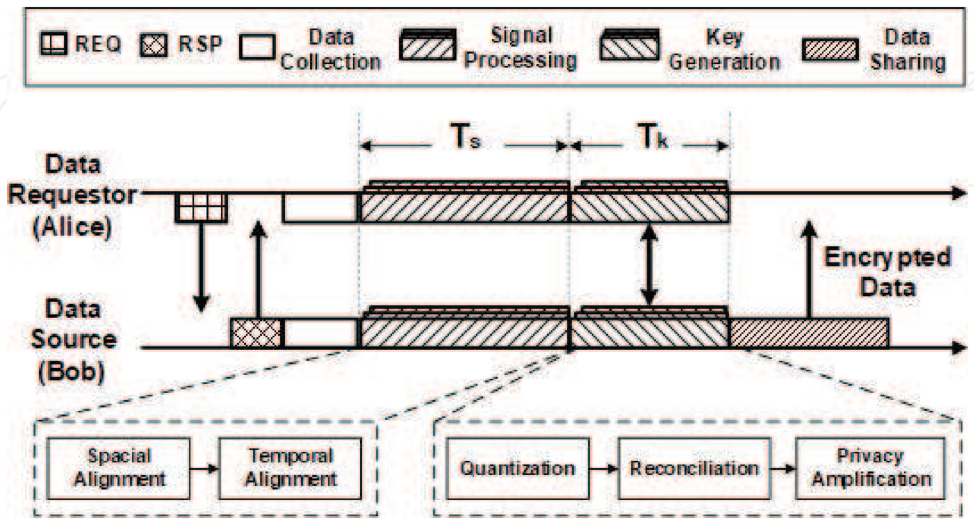

Figure 3. System flowchart.

2. Key generation. The key generation phase is composed of three parts: quantization, reconciliation and privacy amplification. In quantization stage, Alice and Bob convert their observed information into bits. In reconciliation stage, Alice and Bob exchange some information to agree on the same bit string. Finally, privacy amplification technique is used to diminish the partial information eavesdropped by Eve.

In the following sections, we will describe design details of each component in turn.

3.1. Signal processing

In this section, we describe spatial alignment which is used to transform the original acceleration data into the same coordinate system and temporal alignment which addresses the problem of synchronization.

3.1.1. Spatial alignment

Walking is inherently a three-dimensional movement. 3D acceleration data cannot be directly used for key generation because they are recorded at different locations and orientations. We address this problem by transforming accelerometer data of different devices to the same reference coordinate system. **Figure 4** shows the definition of device coordinate system, the global coordinate system, and the body reference coordinate system. We define the world coordinate system by North, East and the Earth gravity (−G). The device coordinate system is defined as (X, Y, Z). The user plane of motion is defined as forward-sideway plane that is perpendicular to gravity. Sideway points to the right side of the body's forward direction. Take smart watch as an example, assume the linear acceleration signals along three orthogonal directions of smart watch are Acc_x, Acc_y and Acc_z, respectively, the linear acceleration in the body reference system can be calculated by the following equation:where Acc_X′, Acc_Y′ and Acc_Z′ are linear acceleration along gravity direction, forward direction and sideway direction in the body reference system. *R* is the rotation matrix from the device coordinate

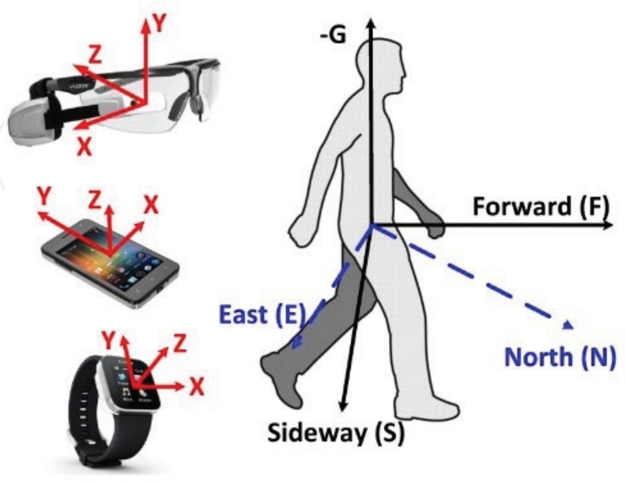

Figure 4. The different coordinate systems: the coordinate system of device is defined as (X, Y, Z). The world coordinate system is defined as (E, N, −G). The user plane of motion is the forward-sideway (F-S) plane, which is the plane perpendicular to gravity.

system to the world coordinate system and can be obtained by the method in [5]. The authors in [5] proposed a method to reliably and accurately estimate the user orientation in different environments at arbitrary phone positions and orientations. Their proposed method first fuses different phone inertial sensors to obtain the phone orientation and then applies principal component analysis (PCA) to improve the final accuracy. After obtaining the acceleration in body coordinate system, we use Acc_X′, Acc_Y′ and Acc_Z′ for key generation.

$$\begin{pmatrix} Acc\_X' \\ Acc\_Y' \\ Acc\_Z' \end{pmatrix} = R \begin{pmatrix} Acc\_x \\ Acc\_y \\ Acc\_z \end{pmatrix} \tag{1}$$

3.1.2. Temporal alignment

Temporal synchronization is necessary because different devices sample acceleration values independently. We adopt an event-based mechanism in which devices detect the time point of a heel-strike event, and use this event to trigger data collection. This method is based on the fact that the acceleration data along gravity direction reach the peak simultaneously when the foot touches the ground, and time delays in signal transmission through the body are negligible. In order to detect this event, we apply a low-pass filter on acceleration data Acc_X′ to reduce noise. The cut-off frequency is 3 Hz as we find the normal step frequency lies between 1.6–2.8 Hz [6]. Then the local peaks are detected to indicate heel-strike events as we can see from **Figure 5**.

The proposed method eliminates the requirement of explicit synchronization as heel-strike events can be detected locally at each device without communication. When Alice receives a *ACK* message from Bob, they both reach an agreement to record acceleration data from the next *i*-th heel-strike event and stop recording at the subsequent *j*-th event. Then the acceleration samples are processed by spatial alignment to the same coordinate system.

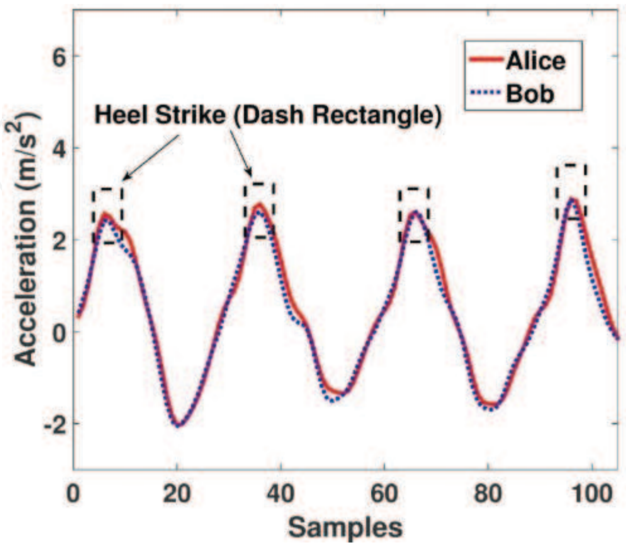

Figure 5. The peak along gravity direction indicates a heel strike on the ground.

3.2. Key generation

3.2.1. Quantization

Our goal is to extract exactly the same cryptographic key if and only if two devices are on the same body. **Figure 6** plots the acceleration of two legitimate devices and an adversary device in three directions. Due to the fact that acceleration values of two devices on the same body are not identical as shown in **Figure 6**, the key generation algorithm must have the ability to map even similar sequences to the same key and less similarity to different keys. The key generation method is applied on two legitimate devices separately.

After signal alignment, we obtain acceleration values along three directions: Acc_X′, Acc_Y′ and Acc_Z′. To generate keys, we perform low-pass filtering and quantization for the acceleration values in these three directions independently. As the first step, we use a low-pass filter to filter out environmental noise. The cutoff frequency is chosen as 10 Hz as the useful frequency of human motion lies below 10 Hz. After filtering, the acceleration values are normalized to have zero mean and unit length to eliminate the impact of different positions. Bits are obtained by applying the bit generation approach in [7, 8]. In the proposed system, we segment the acceleration sequence with a moving window with no overlap (window size $W = 10$). We then calculate two thresholds $q+$ and q- within each window:

$$q+ = u + \alpha * \delta, \quad q- = u - \alpha * \delta \tag{2}$$

where u and δ are the mean and standard deviation of acceleration values in a window. After obtaining $q+$ and $q-$, we generate bits by the following rule: the acceleration values which are > q + are encoded as bit 1, and values that are < q_- are encoded as bit 0. Finally, we combine the bits generated from each window together to form a bit string. **Figure 7** illustrates the quantization process in a window in detail.

After the above steps, we obtain three separate bit strings K_G, K_F, K_S. We concatenate these three bit strings together to form the initial key for Alice K_Alice = [K_G, K_F,K_S]. On Bob's side, he will perform the same quantization process to get K_Bob.

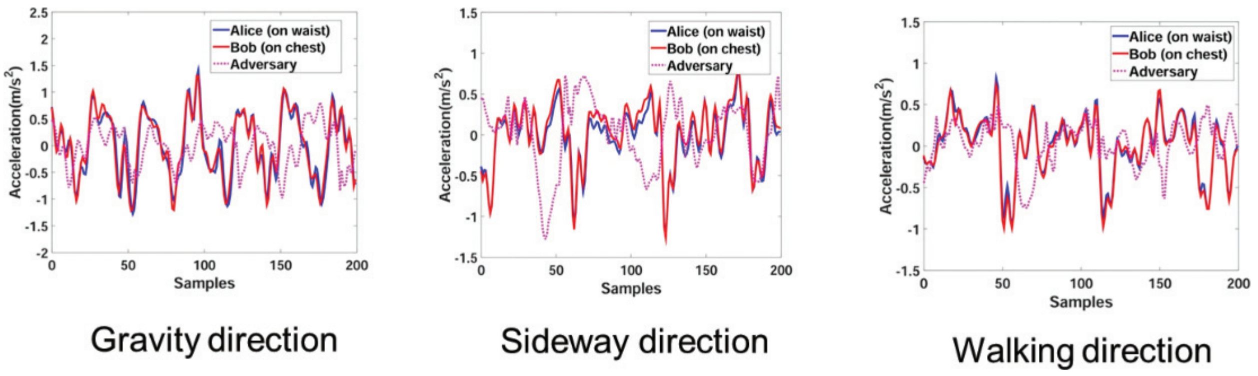

Figure 6. Acceleration along three directions.

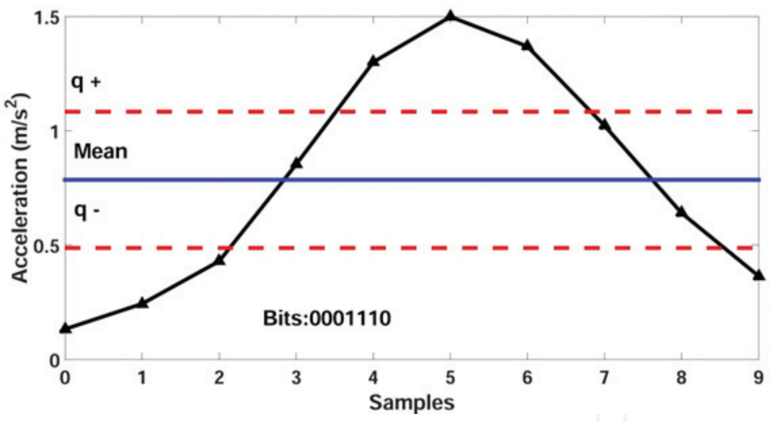

Figure 7. Quantization process.

3.2.2. *Reconciliation process*

After key generation, we often get K_Alice≈K_Bob in practice because of noise. In this section, we describe reconciliation process which is used to correct the mismatches between Alice and Bob. The main reason causing mismatches is that the samples discarded during quantization at Alice's side may be different from Bob's side. In order to correct the mismatches, they are required to exchange their sequence of sample indexes during reconciliation process. In this step, only the bits corresponding to common sample indexes are reserved by Alice and Bob to obtain the common key. For example, let us assume Alice and Bob each has 10 measurements in a window. After key generation, Alice obtains "01100" and Bob yields "0110". The length of these two keys is not required to be same because they may discard different number of acceleration samples during quantization. If we assume Alice generates bits at positions 1, 3, 5, 6 and 9, respectively. She sends L_Alice = [1, 3, 5, 6, 9] to Bob. Bob checks his own position information and finds the bits are generated at positions 1, 2, 5 and 9. He sends L_Bob = [1, 2, 5, 9] back to Alice. Then they use the bits at positions 1, 5 and 9 to generate the same key as K_Alice' = K_Bob' = "010."

3.2.3. *Privacy amplification process*

Because walking is a repetitive activity, different steps may have high correlation, thus decreasing the randomness of the key. This problem can be addressed by privacy amplification process. In the proposed system, we use XOR function to achieve this goal. Specifically, we interleave the bit streams from three directions in time sequence and segment the concatenated keys into small windows with no overlap. We then perform XOR operation on two consecutive windows together to obtain the final key which are denoted by K_Alice" and K_Bob".

Another function of privacy amplification is that it diminishes the partial information revealed to Eve. Because in the reconciliation stage, Alice and Bob exchange messages over an authenticated public channel and the publicly exchanged messages reveal a certain amount of information about the bit strings to Eve. Note that other privacy amplification methods such as universal hash [9] can be employed to further enhance the randomness of the concatenated key. We refer the reader to [9] for more details.

http://dx.doi.org/10.5772/intechopen.74195

4. Performance evaluation

In this section, we present the evaluation results. The aims of the evaluation are twofold: (1) to evaluate the impact and performance of different parameters such as the window size (W), α in quantization process, sampling frequency (F_s); (2) to evaluate the performance of different components in the system which includes reconciliation process and privacy amplification.

4.1. Data acquisition

We collected a set of accelerometer data from 20 subjects (14 males and 6 females) wearing smart devices on different locations of their body. As can be seen from **Figure 8**, the body positions involved in the data collection include head, chest, waist and wrist. These positions represent the common locations of mobile devices such as smartwatch and medical sensors (e.g. pacemaker). During the data acquisition, the sampling rate of all smart wearable devices is 100 Hz.

During data acquisition, the participants were asked to wear mobile devices as shown in **Figure 8** and walk for about 5 minutes in their normal speed (0.7–1.1 m/s). We collect data from both indoor and outdoor environments to capture different terrains in real-world scenarios. It is worth mentioning that we do not consider the influence of different days or different walking speeds

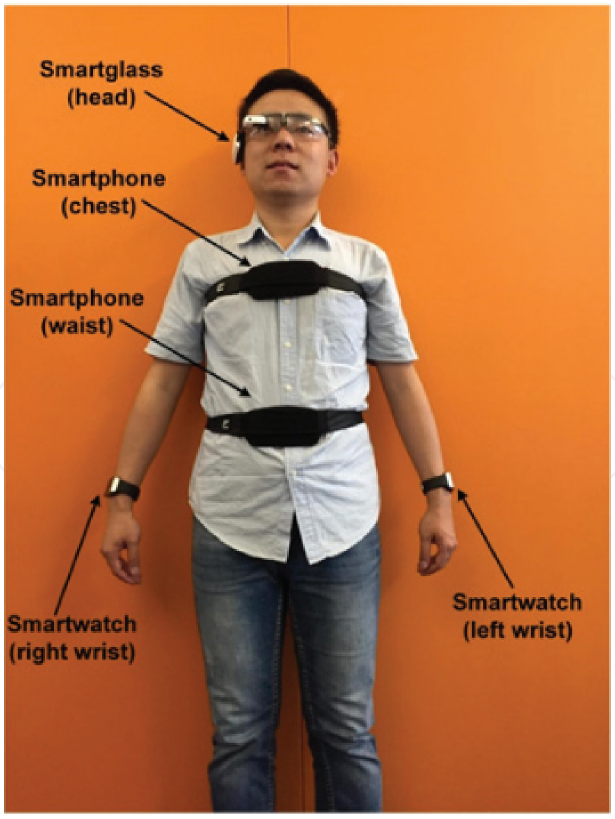

Figure 8. Data acquisition.

(slow, normal and fast) because all the devices worn by the subject are measuring the same gait signal simultaneously, which is different from the data collection requirement in the study of gait recognition [10]. The detected peaks that indicate heel strikes are used to synchronize acceleration samples recorded on different wearable devices and segment steps. For each device attached on one subject, we segment the acceleration sequences into small windows based on heel-strike points. In this experiment, each window contains 10 steps. Then these windows are passed to the system to generate final keys and evaluate the following metrics.

Three metrics are selected to quantitatively evaluate the performance:

- **Bit agreement rate**. Bit agreement rate denotes the percentage of matching bits in keys generated by two parties. This metric evaluates the potential that two parties (either two legitimate devices or a legitimate and an adversary device) can generate the same key.
- **Bit rate.** Bit rate denotes the number of bits generated in unit time, measured in bits per second (bps). This metric evaluates how fast the system can generate a secret key.
- **Entropy.** Entropy denotes randomness in generated keys. This metric measures the amount of information contains in each bit.

To evaluate the influence of different parameters on the system performance, we first conduct a systematic exhaustive search to find the optimal combination. We vary the respective parameters within a dedicated range, i.e. $F\_s = 10, 20, 30, 50, 100$, $\alpha = 0, 0.1, \ldots, 1$, and $W = 5, 10, \ldots, 50$. After this step, we find the best parameters combination is F_s = 30, $\alpha = 0.8$ and $W = 10$. Then we take turns to evaluate the influence of each parameter on system performance by setting the other two parameters to the optimal value.

As mentioned above, each subject has acceleration samples recorded from five different body parts. For each body part of one subject, we compute the bit agreement rate of legitimate devices by using keys generated from other locations on the same body. We show the results of the average values and 95% confidence level of the performance metrics (bit agreement rate and bit rate).

4.2. Influence of sampling rate

In this experiment, we evaluate the influence of different sampling rate by downsampling F_s from 100 to 50 Hz, 30, 20 and 10 Hz, respectively. The impact of F_s on bit rate and bit agreement rate are plotted in **Figure 9**, respectively. We find that sampling rate has a negative impact on the agreement rate between legitimate devices. After looking into the data, we find this is because a higher sampling rate is able to record more acceleration values during the same time window and thus increase bit rate; however, it reduces bit agreement as a higher sampling rate captures acceleration variation more precisely which results in more mismatches between legitimate devices.

4.3. Influence of parameter α

The parameter α in quantization phase determines the trade-off between agreement rate and bit rate. In this experiment, we will evaluate the influence of α. As we can see from **Figure 10**,

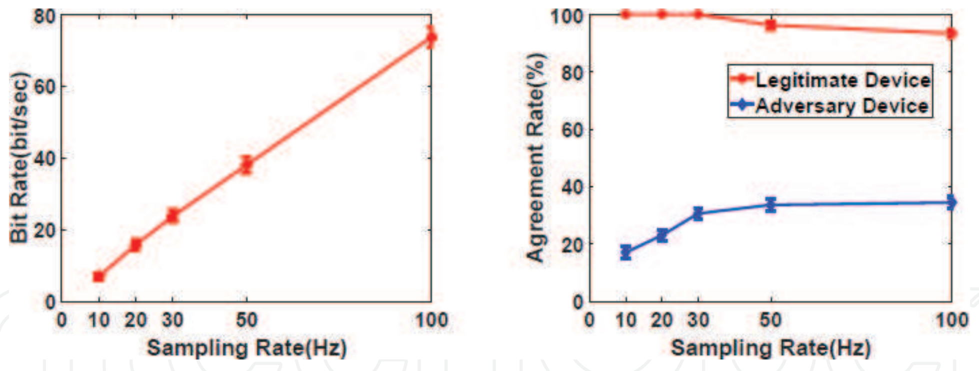

Figure 9. Influence of sampling rate.

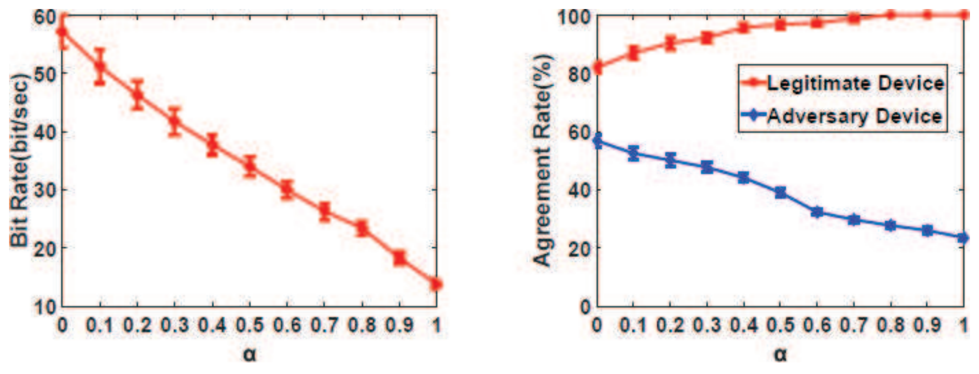

Figure 10. Impact of α.

the bit rate drops when α increases. This is because the parameter α in Eq. (2) determines the guard band which affects the inclusion or exclusion of acceleration samples. A larger α means we will discard more acceleration samples. This will reduce the bit rate. On the other hand, as we can see from **Figure 10**, the bit agreement rate increases with the growth of α because more mismatches are excluded. In the meantime, we observe that parameter α has inverse impact on agreement rate for an adversary device. This is because more samples are discarded for quantization at all the devices when α increases. Therefore, the legitimate devices know the index numbers used and they exchange the index list during reconciliation, so the agreement rate increases. However, for an attacker, she does not know which samples are kept because the signal values itself are different from that of legitimate devices, the remaining bits after discarding more acceleration measurements in quantization will have less bit agreement rate.

In addition to sampling rate and α, we also evaluated the influence of window size in the proposed system. We found that the moving window size W does not have much influence on the system performance and we set the moving window size to 10 for the proposed system.

4.4. Influence of reconciliation process

As described in Section 3.2.2, reconciliation process is necessary because it is used to reduce the mismatches between two parties. **Figure 11** shows the influence of reconciliation process on the bit rate and agreement rate. We can see a significant improvement in bit agreement rate

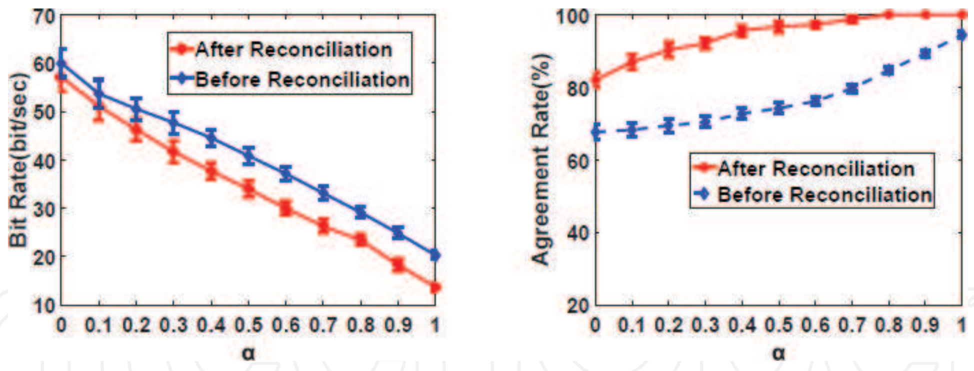

Figure 11. Influence of reconciliation.

after employing reconciliation approach. However, we also notice that the drawback of reconciliation approach used in our system is that it will reduce bit rate. Recall that the main goal of our system is to generate the same key; therefore, reconciliation is a necessary part of the proposed key extraction system.

4.5. Impact of privacy amplification

As mentioned in Section 3.2.3, privacy amplification is used to improve the randomness of the final key. In this experiment, we investigate if the XOR function in privacy amplification achieves this goal. **Figure 12** compares the entropy of the final key before and after privacy amplification. We find that the distribution of entropy is closer to 1 after XOR operation. We also notice that the entropy of the final keys varies from 0.93 to 1 which suggests that the proposed key generation protocol can generate secret keys with good entropy. It is worth mentioning that one disadvantage of using XOR function is that the bit rate will be reduced by a factor of 2 because we XOR two consecutive windows together. From the results in **Figure 11**, we find that the final bit rate can still reach 26 bit/sec after reconciliation and privacy amplification process. State-of-the-art cryptographic algorithms such as AES require a key

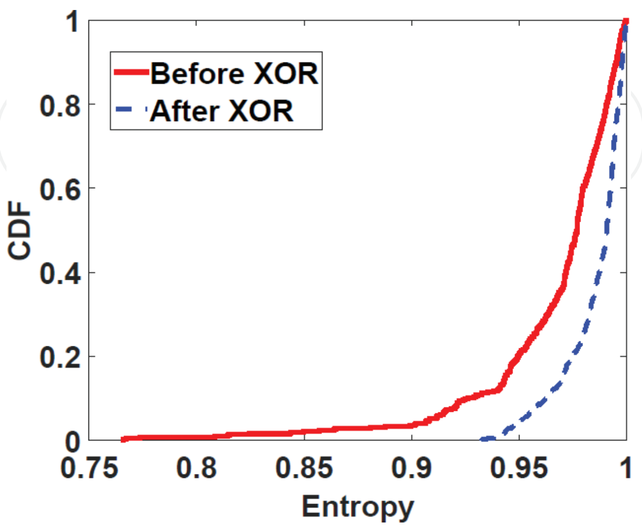

Figure 12. Impact of privacy amplification.

NIST test	p-values
Frequency	0.606072
FFT test	0.562699
Longest run	0.027173
Linear complexity	0.386887
Block frequency	0.984496
Cumulative sums	0.974180
Approximate entropy	0.995898
Nonoverlapping template	0.302941

Table 1. NIST test results.

length with at least 128 bits; therefore, our method takes about 5 s to generate a secure key (about 10 steps).

4.6. Evaluation of key randomness

The generated keys will be used for encryption and decryption; therefore, we need to guarantee that the generated keys are random. In this experiment, we apply the NIST suite of statistical tests to evaluate the randomness of the final keys generated from our dataset. The final results of NIST statistical test are p-values of different random test processes. If p-values are greater than 0.01, it indicates the bit string is generated by a random process. Otherwise, if p-values are below 0.01, the key is not random. From **Table 1**, we can see that the p-values are all greater than 1% which indicates the generated keys from our method are random.

5. System implementation

In order to demonstrate the feasibility of the proposed key generation protocol on mobile devices, we have prototyped the proposed scheme on Motorola Moto E2, a state-of-the-art mobile phone. The complete key generation scheme is implemented in Java. The sampling rate of accelerometer is set as 30 Hz, and Bluetooth low energy (BLE) functionality is employed for wireless communication. BLE is designed to provide significantly lower power consumption for devices with low power requirements, such as pacemaker. Therefore, BLE is well suited for the proposed key generation protocol. Moreover, devices working under BLE broadcasting mode do not need to be paired together beforehand.

BLE is designed to provide significantly lower power consumption for devices with low power requirements. It introduces a new feature called peripheral mode, in which the data source can advertise and publish data without requiring to pair with the data requestor beforehand. BLE peripheral mode is designed for devices with resource-constraint and need to publish new data frequently. Therefore, we run the system in peripheral mode and advertise the data using broadcast packets. Bob organizes its data using Generic Attribute Profile (GATT) and encrypts

	Computation time (ms)	Energy consumption(mJ)
Signal processing	108.3	72.7
Key generation	208.1	12.7
AES encryption	0.2	0.1
AES decryption	0.2	0.1
Total	316.8	85.6

Table 2. System overhead measured on Moto E2 smartphone.

the data to publish by AES. All the devices nearby including adversaries can receive the broadcast advertisements and read the public-available data from Bob. However, only Alice on the same body can generate the same key for data decryption. In this way, the private data is protected from reading by unauthorized devices.

We implement our protocol on two Moto E2 smartphones and use one of them to simulate pacemaker (Bob), the other one is used as smartphone (Alice) to request data from pacemaker. Both Alice and Bob work under BLE broadcasting mode. The main interface of Alice has a UI button that can be pressed by a user to trigger pairing process. After a user presses the start button on Alice, Alice and Bob follow the steps presented in **Figure 3** to generate a shared secret key. If the length of the final key is greater than 128, only the first 128 bits are considered. After obtaining a 128-bit length cryptographic key, Bob encrypts the heart rate measurement (simulated measurement) by Advanced Encryption Standard (AES) and broadcast the encrypted message. Although several nearby devices including adversaries can listen to the broadcast packets, only Alice can generate the same key and decrypt the messages.

Table 2 presents the system overhead (computation time and energy consumption) of our system on Moto E2. The computation time and energy consumption are computed by averaging the results from 50 trials. The computation time is obtained from the console of Android studio. The energy consumption is profiled by $E = PT$, where P is the average power and T is the running time of the profiled component. The average power $P = Current *Voltage$, where the *Current* and *Voltage* of the battery is obtained via Android APIs. The results in **Table 2** show that the execution of the two stages in the protocol: signal processing and key generation take an average time of 108.3 and 208.1 ms, respectively. When the scheme is fully employed, the computation time and energy consumption are 316.8 ms and 85.6 mJ, respectively. Note that heel-strike detection is applied on the continuous acceleration samples to detect heel-strike events. Therefore, the processing time and energy consumption are not applicable in **Table 2**. These results demonstrate the proposed key generation protocol has a low system overhead and can run in real time on modern mobile devices.

6. Conclusion

In this chapter, we propose and implement a key generation protocol that exploits the acceleration data produced by walking to establish a common cryptographic key between two wearable devices. Our protocol obtains a security advantage from the fact that different people have distinctive walking styles. We performed extensive experiments to evaluate the performance of our proposed system and demonstrate that the proposed key generation protocol is able to establish a 128-bit length key in about 5 s (about 10 steps). We also prototyped the proposed scheme on Motorola E2 smartphone to demonstrate the feasibility of our system on mobile devices.

Author details

Weitao Xu* and Guohao Lan

*Address all correspondence to: w.t.xu@unsw.edu.au

School of Computer Science and Engineering, University of New South Wales, Sydney, Australia

References

[1] Stajano F, Anderson R. The resurrecting duckling: Security issues in ad-hoc wireless networks. In: Proceedings of the 7th International Workshop in Security Protocols, Lecture Notes in Computer Science. London, UK: Springer-Verlag; 1999. http://www.cl.cam.ac.uk/fms27/duckling

[2] Gehrmann C, Mitchell CJ, Nyberg K. Manual authentication for wireless devices. RSA Cryptobytes. 2004;**7**(1):29-37

[3] Die W, Hellman ME. New directions in cryptography. IEEE Transactions on Information Theory. 1976;**22**(6):644-654

[4] Gafurov D, Snekkenes E, Buvarp TE. Robustness of biometric gait authentication against impersonation attack. In: Proceedings of on the Move to Meaningful Internet Systems 2006 (OTM 2006 Workshops). Berlin, Heidelberg: Springer; 2006. pp. 479-488

[5] Mohssen N, Momtaz R, Aly H, Youssef M. It's the human that matters: Accurate user orientation estimation for mobile computing applications. In: Proceedings of the 11th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services (MOBIQUITOUS). ICST (Institute for Computer Sciences Social-Informatics and Telecommunications Engineering), London, United Kingdom; 2014. pp. 70-79

[6] Murray MP. Gait as a total pattern of movement: Including a bibliography on gait. American Journal of Physical Medicine & Rehabilitation. 1967;**46**(1):290-333

[7] Revadigar G, Javali C, Hu W, Jha S. DLINK: Dual link based radio frequency fingerprinting for wearable devices. In: Proceedings of IEEE Conference on Local Computer Networks (LCN). Clearwater Beach, FL, USA: IEEE; 2015. pp. 154-164

[8] Javali C, Revadigar G, Ding M, Jha S. Secret key generation by virtual link estimation. In: Proceedings of EAI International Conference on Body Area Networks (BodyNets). ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering). Brussels, Belgium: ICST; 2015. pp. 63-72

[9] Bennett CH, Brassard G, Robert J-M. Privacy amplication by public discussion. SIAM Journal on Computing. 1988;**17**(2):210-229

[10] Ren Y, Chen Y, Chuah MC, Yang J. Smartphone based user verification leveraging gait recognition for mobile healthcare systems. In: Proceedings of Sensor, Mesh and Ad Hoc Communications and Networks (SECON). New Orleans, LA, USA: IEEE; 2013. pp. 149-157