InaudibleKey: Generic Inaudible Acoustic Signal based Key Agreement Protocol for Mobile Devices

Weitao Xu
weitaoxu@cityu.edu.hk
City University of Hong Kong

Zhenjiang Li
zhenjiang.li@cityu.edu.hk
City University of Hong Kong

Wanli Xue
wanli.xue@cybersecuritycrc.org.au
University of New South Wales*

Xiaotong Yu
xiaotong.yu96@gmail.com
University of New South Wales

Bo Wei
bo.wei@northumbria.ac.uk
Northhumbria University

Jia Wang
jia.wang@szu.edu.cn
Shenzhen University

Chengwen Luo
chengwen@szu.edu.cn
Shenzhen University

Wei Li
weiwilson.li@sydney.edu.au
The University of Sydney

Albert Y. Zomaya
albert.zomaya@sydney.edu.au
The University of Sydney

ABSTRACT

Secure Device-to-Device (D2D) communication is becoming increasingly important with the ever-growing number of Internet-of-Things (IoT) devices in our daily life. To achieve secure D2D communication, the key agreement between different IoT devices without any prior knowledge is becoming desirable. Although various approaches have been proposed in the literature, they suffer from a number of limitations, such as low key generation rate and short pairing distance. In this paper, we present InaudibleKey, an inaudible acoustic signal based key generation protocol for mobile devices. Based on acoustic channel reciprocity, InaudibleKey exploits the acoustic channel frequency response of two legitimate devices as a common secret to generating keys. InaudibleKey employs several novel technologies to significantly improve its performance. We conduct extensive experiments to evaluate the proposed system in different real environments. Compared to state-of-the-art works, InaudibleKey improves key generation rate by 3-145 times, extends pairing distance by 3.2-44 times, and reduces information reconciliation counts by 2.5-16 times. Security analysis demonstrates that InaudibleKey is resilient to a number of malicious attacks. We also implement InaudibleKey on modern smartphones and resource-limited IoT devices. Results show that it is energy-efficient and can run on both powerful and resource-limited IoT devices without incurring excessive resource consumption.

KEYWORDS

Key generation, Mobile devices, Acoustic signal

1 INTRODUCTION

1.1 Background

With recent advances in mobile computing and embedded technology, there is an increasing number of IoT devices in our daily life, such as smartphone, smart watch, and Google assistant. Correspondingly, it is more and more common to pair two devices for the purpose of data sharing, synchronization, and collaboration. For example, two persons that meet for the first time in a meeting want to associate their smartphones temporarily to exchange their name card. Due to the "open air" nature of wireless communication, cryptographic key agreement is a fundamental requirement to secure D2D communication to achieve confidentiality [1, 2].

The secure key distribution between two communication parties can be addressed by public key infrastructure (PKI). Unfortunately, public key-based solutions are not applicable to mobile devices because PKI only works if the identity of the other party is known out of band or only trusted parties have identities signed by pre-established certificate authorities. Another solution is pre-distributed key which is usually in the form of master key or key material. However, key pre-distribution schemes lack scalability, which makes them incapable in a dynamic environment, where new devices may join and quit frequently. Near field communication (NFC) is becoming more and more popular in modern mobile devices, but its communication range is limited to only tens of centimetres (typically <20 cm). Diffie-Hellman protocol (also named D-H protocol) is a popular key establishment protocol to establish cryptographic keys over a public channel. However, D-H protocol is susceptible to man-in-the-middle (MITM) attack and authenticated D-H protocol requires the presence of certificate authority (CA). The most common method to pair mobile devices is still asking the user to scan nearby devices, choose the target device and confirm manually, which is neither user friendly nor suitable for devices without screens.

1.2 Motivation

The lack of efficient and user-friendly pairing methods has inspired researchers to explore suitable alternatives to authenticate mobile devices. Because two devices are not assumed to own any shared knowledge as a-priori, the widely adopted design principle in the literature is that if multiple devices share a similar observation of certain random signal, then the signal can be used to extract keys.

IPSN '21, May 18-21, 2021, Tennessee, USA

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

Different designs explore different forms of such random signals [3–11], while they all strive to achieve a *fast* (sufficient bit generation rate), *practical* (without requiring extra hardware) and *ubiquitous* (usable in different environments) key generation system design.

A successful set of pioneer efforts is to leverage wireless channel information [1, 3–6, 12], such as Received Signal Strength Indicator (RSSI) and Channel State Information (CSI). These methods are based on wireless channel reciprocity which means the channel characteristics (RSSI or CSI) measured between two devices by exchanging a pair of probe packets in quick succession, will be nearly the same. However, RSSI-based methods suffer from low key generation rate and predictable channel attack [3, 5]. CSI-based systems can improve key generation rate greatly and are robust to predictable channel attack [4]. However, the major limitation is it requires special toolkits to extract CSI from wireless card, and currently CSI can only be extracted from a limited number of chipsets such as Intel's 5300 NIC [13, 14].

To overcome this issue, many recent designs appear for mobile platforms by utilizing the on-board sensory data, such as acoustic data, motion sensor data, bio-sensor data, etc. However, through our study, we find that they mainly trade the sensor's availability for other two further restrictions — either the system works in a very short pairing distance, e.g., 1.25 cm in Proximate [8], 5cm in TDS [4], or in a limited set of pre-defined contexts, e.g., in certain environments, when users are performing certain activities [10, 11], etc. The designs requiring short pairing distance are not preferable because people need to keep social distance (usually >1.5 m) in their daily life since the outbreak of COVID-19. Many other approaches may further suffer a long authentication delay [15], need expensive software to support (e.g., public key) [16], or require extra hardware (e.g., bio-sensors) [17, 18]. In this paper, we thus aim to investigate whether we can still achieve a fast and ubiquitous key agreement system design that can pair two mobile devices beyond social distance, yet without using extra sensors not available on mobile devices?

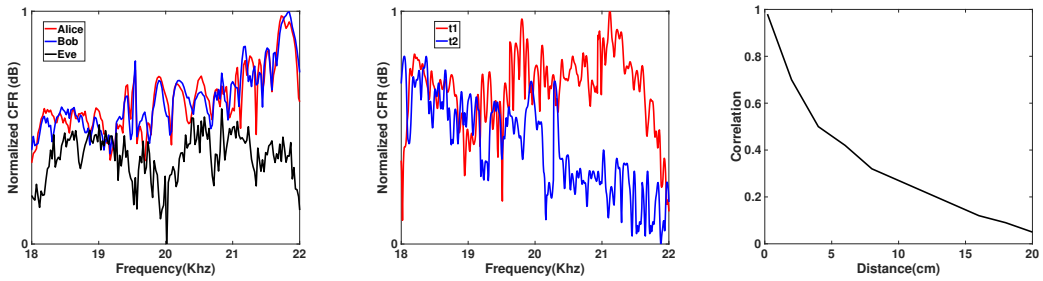
1.3 Our Approach and Design Challenges

We find that the acoustic signals have such a great potential, inspired by the success from previous radio signal based designs. Acoustic wave, as a form of wave, possesses many properties that radio wave has. In particular, we exploit to leverage *acoustic channel reciprocity* to generate keys. One primary advantage of using acoustic signal is that it is not dependent on special hardware as most mobile devices are equipped with microphone and speaker. Our preliminary study also validates that the acoustic channel indeed holds channel reciprocity, as well as temporal variation and spatial decorrelation, which could serve as the basics of key generation. However, because of the limited acoustic channel bandwidth and the location offset between speaker and microphone, a number of challenges need to be addressed to design an efficient and robust key agreement protocol based on the acoustic signal.

(1) Firstly, how to achieve high bit generation rate through narrow band acoustic channels. To this end, we need to extract fine-grained acoustic channel information. Unfortunately, different from wireless card, microphone is not designed to provide channel information such as RSSI and CSI. To extract fine-grained channel information, we design an effective transmitting scheme that uses the inaudible acoustic signal to modulate Orthogonal Frequency-Division Multiplexing (OFDM) symbols. Although applying OFDM modulation in acoustic signals is proposed in FingerIO [19], the use of OFDM in a key generation system can provide more acoustic channel information that can be used to generate keys. As a result, the key generation rate is significantly improved as demonstrated in our evaluation.
(2) Secondly, how to further improve the entropy of the extracted key. Existing quantization methods usually use a threshold to determine whether a sample should be encoded to '1' or '0' [3]. However, these methods may produce repeated bit strings which reduce the entropy of the generated keys. Additionally, these keys are used directly to generate the final key, which leaves opportunities for powerful attackers to obtain raw information by reverse engineering. To tackle this problem, we first apply a novel Bloom filter-based technology to protect the keys against reverse engineering attack. Then, we leverage Karhunen-Loeve Transform (KLT) to remove the redundant information and enhance randomness.
(3) Thirdly, the microphone and speaker are not located at the same location in mobile devices. Hence, the transmitted signal and received signal will experience slightly different channels. Moreover, due to hardware diversity and manufacture imperfection [20], different microphones/speakers attenuate some frequencies selectively which will further cause more errors. To address this challenge, we optimise a novel compressive sensing (CS) based reconciliation mechanism. The discrepancies between two initial keys can be corrected with the help of powerful ℓ_1 optimization. In particular, InaudibleKey can achieve high matching rate even for different types of IoT devices.

Although several recent works have exploited acoustic signal to pair mobile devices [15, 16, 21, 22], our system shows significant performance improvement. We make the following contributions in this paper:

- **System Design.** We propose InaudibleKey, an inaudible acoustic signal-based key agreement protocol for mobile devices. Based on acoustic channel reciprocity, InaudibleKey utilizes the channel frequency response of OFDM symbols to generate keys. InaudibleKey employs several novel approaches to significantly improve the system performance. Particularly, we propose an optimisation algorithm to improve the performance of a state-of-the-art reconciliation method.
- **System Implementation.** To demonstrate the feasibility, we implement the prototype of InaudibleKey on both powerful devices (smartphone) and resource-limited devices (Arduino Uno board). Evaluation results show that InaudibleKey incurs low system cost and can run efficiently on these IoT devices. We also demonstrate that it is more energy efficient than public key cryptography and authenticated D-H protocol on IoT devices.
- **System Evaluation.** We conduct extensive experiments in different real environments. Compared to state-of-the-art works,



(a) Reciprocity (b) Temporal variation (c) Spatial decorrelation

Figure 1: Feasibility Study.

InaudibleKey improves key generation rate by 3–145 times, extends pairing distance by 3.2–44 times, reduces information reconciliation counts by 2.5–16 times.

- **Security Analysis.** Extensive analysis shows that InaudibleKey is robust to a number of malicious attacks, such as eavesdropping attack, imitating attack and predictable channel attack.

The rest of the paper is organized as follows. We present preliminary study results in Sec. 2, followed by a description of the system model in Sec. 3. Then we specify the system design in Sec. 4. We evaluate the performance of InaudibleKey in Sec. 5 and analyze its security against attacks in Sec. 6. Finally, Sec. 7 discusses related works, and Sec. 8 concludes the paper.

2 FEASIBILITY STUDY

We first conduct preliminary study to verify whether acoustic channel hold channel reciprocity, temporal variation and spatial decorrelation which serve as the basis for key generation.

Channel reciprocity. Channel reciprocity means the channel characteristics (gains, phase shifts, and delays) measured between two devices by exchanging a pair of probe packets within channel coherence time will be very close [3]. For wireless channel, the widely used channel characteristics include RSSI [3, 23] and CSI [1, 4, 6]. Unfortunately, the microphone cannot report such channel information. Recent studies use channel taps [1] [21] and sound pressure [22] as acoustic channel characteristics. However, we find that they can only provide coarse-grained channel measurements.

In this paper, we use channel frequency response (CFR), which can provide fine-grained acoustic channel information. CFR means the response of a channel at different frequencies. The multipath fading affects different frequencies across the channel to different degrees giving rise to frequency selective channel [24]. Moreover, if one of the devices is moving such as shaken by the user, it will also cause channel variations due to Doppler effect. Therefore, the received signal will have different responses at different frequencies. In other words, the randomness in acoustic channel responses can be used to generate keys just like radio channel characteristics.

To validate the channel reciprocity, Alice and Bob exchange a number of acoustic signals while Eve (20 cm away) is eavesdropping the acoustic signal between Alice and Bob. Fig. 1(a) plots the CFR of Alice, Bob and Eve. We can see that the CFR of the legitimate devices are close to each other while Eve has different channel responses (though there are some similarities in certain frequencies). However, we also find the CFRs of Alice and Bob are not exactly the same. First, the signals transmitted by Alice and Bob are not transmitted exactly along the same path because the speaker and microphone are located in different locations on the smartphone. This is different from the wireless devices transmission, where the transceivers could always use one specific antenna. Second, the hardware frequency selectivity from imperfect hardware such as speaker [20], can also result in the CFR difference. Therefore, we can see different strengthened and weakened power amplitudes at each frequency.

Temporal variation. We set up two mobile phones—namely, Alice and Bob— in a laboratory with a distance of 1 m. Alice keeps sending inaudible acoustic signals whose frequency varies from 18 kHz to 22 kHz, while Bob is listening to the acoustic channel. Fig. 1(b) shows the channel frequency response at two different times (t1 and t2 in Fig. 1(b)) when the user shakes one of the devices. We can see the acoustic channel response at different time instances are different. When the user is shaking device randomly the acoustic channel between Alice and Bob changes rapidly. If the environment changes such as a person walks by, the acoustic channel will also change due to multi-path and Doppler effect just like the radio channel.

Spatial decorrelation. To validate spatial decorrelation, we vary the distance between Eve and Bob from 1 cm to 100 cm. As shown in Fig. 1(c), the correlation between Eve's and Bob's channel responses decreases rapidly as the distance increases. According to radio propagation theory [24], the channel will be statistically uncorrelated if two devices are separated by half wavelength away. If we use 22 kHz frequency audio signal, the wavelength λ is 1.7 cm. Therefore, if Eve is located away from a legitimate device by $\lambda = 0.85$ cm, it will have different channel measurements. In practice, however, this distance is set as at least multiple wavelengths to alleviate a poor multipath scattering environment or interference and enhance security [25]. From Fig. 1(c), we can see that the correlation drops below 0.4 when the distance between Eve and legitimate device is greater than 10 cm. Hence, the secure distance is 10 cm in this paper, and we assume any attacker entering this range can be easily spotted by Alice or Bob.

3 SYSTEM MODEL

Fig. 2 illustrates the system model in this paper. We assume that two mobile devices, namely Alice and Bob, intend to generate the same

[1] channel taps are the aggregate of delays caused by multi-path effect [24].

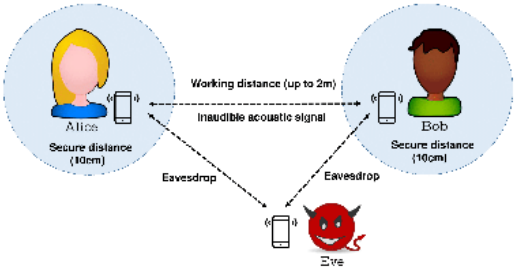


Figure 2: System model

key to secure their communication. Both devices are equipped with a speaker and a microphone. They have no prior shared secrets except that they have InaudibleKey installed. We assume that an adversary device Eve is located beyond a safe distance (10 cm in InaudibleKey) to the legitimate devices. If Eve moves within the safe distance, it can be easily spotted by the users as noted in [16]. One application scenario can be illustrated as follows. Suppose in a meeting, Alice and Bob who meet for the first time want to exchange their name card safely. But to keep social distance, they cannot establish a secure communication channel via existing approaches, such as Touch-and-Guard [18], shake-n-shack [11] or NFC. By using InaudibleKey, they can simply shake their device (e.g., smart watch) or perform a random gesture near the device for a short while. A secure communication is then established between them even they are 1-2 m away from each other. If there are sufficient randomness such as moving subjects/objects around users, they do not even need to take any actions (see our demo [2]).

We assume that Eve has the full knowledge of the key agreement protocol and can eavesdrop, inject, and replay messages. However, like many previous key generation studies [1, 4, 10, 16, 21], although Eve can inject messages to the public wireless channel, we assume the goal of Eve is to intercept the secret key rather than jamming their communications (i.e., DOS attack). In fact, the DOS attack against InaudibleKey can be performed by jamming inaudible acoustic signals into the environment. We can use the methods in the literature [26] to detect such attack. So we consider three types of attacks that are commonly used in related work [1, 21].

- Eavesdropping attack: Eve eavesdrops all the messages transmitted in the public channel to extract the same key.
- Imitating attack: After Alice or Bob finish key extraction, Eve approaches the same site with the aim of generating the same key as a legitimate user. For example, Eve can first observe how Alice or Bob uses devices, e.g. the way of moving or shaking smartphones, then try to imitate his/her use pattern and generate the same key.
- Predictable channel attack: Eve can deliberately move around to generate desired or predictable changes in the channel between Alice and Bob.

4 SYSTEM DESIGN

Fig. 3 shows the work-flow of InaudibleKey. Suppose Alice and Bob are two devices that want to generate a secret key. Firstly, they exchange a number of inaudible acoustic frames and calculate the CFR. Then both devices follow the steps in Fig. 3 to generate

[2]https://drive.google.com/file/d/1lQIMuJahDV5PFhucXj3VODuFyXunn2Y5/view

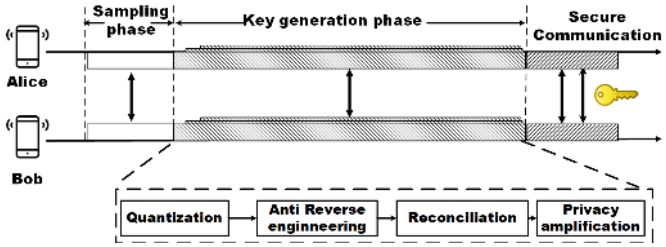


Figure 3: System Flowchart.

the same cryptographic key. Finally, they use the extracted key to secure their communication.

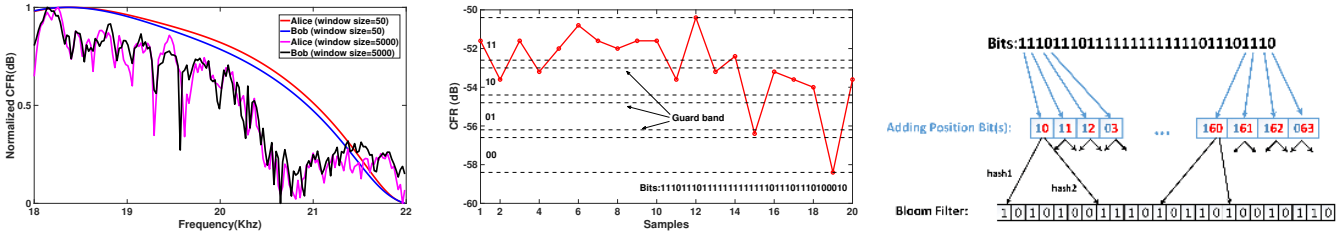
4.1 Transmitting signal design

In sampling phase, both Alice and Bob exchange a number of acoustic frames by transmitting via speaker and receiving via microphone to obtain channel measurements. InaudibleKey uses inaudible frequency band from 18 kHz to 22 kHz for not disturbing users.

To obtain fine-grained channel information, we apply OFDM technology on an acoustic signal based on the method in FingerIO [19]. Specifically, we divide the 18-22 kHz frequency band into 64 subcarriers so that the width of each subcarrier is 62.5Hz. The transmitting time-domain samples can be obtained by performing inverse Fast Fourier transform (IFFT) on the transmitted data, and the receiver can reconstruct the raw data bits by a Fast Fourier Transform (FFT). A speaker will transmit the vectors with 64 real values from the OFDM symbol construction. Another advantage of using OFDM technology is that both devices can probe channel within channel coherence time without explicitly synchronising two mobile devices. In fact, it is impractical to assume two mobile devices are synchronised when they first meet each other. We use the first S_{suf} of these values to form a cyclic suffix that is appended to the end of OFDM symbol. The cyclic suffix is used to accurately estimate the beginning of the OFDM symbol. Even Alice and Bob are not synchronized, they can still locate the beginning of the received symbol by calculating the correlation between the received signal and known transmitting signal (see [19] for more details).

The length of the transmitting signal and the transmitting interval is important for the following two reasons. 1) We need to make sure Alice and Bob obtain the channel estimation within coherence time, so their CFRs are highly correlated; 2) the transmitting interval should be larger than the coherence time. Otherwise, the consecutive CFRs will be correlated and the randomness of the key will be reduced. In theory, the rate at which the channel varies is represented by Doppler frequency (f_d) and the duration when the channel is stable which is denoted by channel coherence time (T_c). Coherence time is the time domain dual of Doppler spread and is used to characterize the time varying nature of the channel frequency. Suppose the moving speed of the subject or object is v, the channel frequency is f, and the speed of acoustic signal is c ($340m/s$), then the maximum Doppler frequency is $f_d = \frac{v \cdot f}{c}$ [27]. Practically, the channel coherence time with respect to the maximum Doppler frequency shift is $T_c = \sqrt{\frac{9}{16\pi f_d^2}}$ according to [27].

The acoustic signal used in InaudibleKey is from 18 kHz to 22 kHz, and the speed of common human motions varies from



(a) Impact of window size (b) An example of quantization (c) Projecting the Bits into Bloom filter space

Figure 4: Illustration of quantization process.

0.1–2.7 m/s [28]. Therefore, the coherence time lies in the range of 2 ms to 53 ms. In InaudibleKey, we set the length of the cyclic suffix S_{suf} to 26. Therefore, the transmitted symbol contains 90 samples. Given 48 kHz sampling rate, the transmitter takes 1.9 ms to transmit these 90 samples, which is shorter than the minimum coherence time. In terms of transmitting interval, Alice and Bob exchange acoustic signals every 100 ms, which is longer than the maximum coherence time.

4.2 Quantization

4.2.1 Multiple-bit quantizer. Upon receiving the signal from Bob, Alice applies a Butterworth band-pass filter (18–22 kHz) to filter out the environmental noises and calculate the CFR using the method in [16]. When calculating CFR, the window length plays an important role: if it is too small, there is not much entropy; however, if it is too large, there will be more mismatches (Fig. 4(a)). We empirically use a Hamming window whose size is 2000 in InaudibleKey.

Then the CFR is quantized to binary bits (0s and 1s) by multiple-bit quantization technique proposed in [3]. To be specific, we first divide the CFR measurements into several windows with no overlap (window size $W = 20$). Then for each window, we divide the samples into multiple quantization levels. Each level in the quantization is assigned an n–bit code. For example, if $n = 2$, then each sample will be converted into 2 bits. We also insert guard band between different levels to mitigate the effect of mismatches. We use $\alpha \in [0, 1]$ to represent the ratio of guard band to data. The larger the α is, the more mismatches are discarded. However, it is possible that the length of the keys generated by Alice and Bob are different. To solve this issue, we exchange the indices of samples that are used to generate keys and only reserve keys generated at the common indices. Fig. 4(b) shows how to convert a window of 20 samples into bits. After quantization, we assume the keys generated by Alice and Bob are denoted by K'_{Alice} and K'_{Bob}.

4.2.2 Anti Reverse Engineering. Most previous work on key generation utilize the quantized bits directly to get the final secret key. However, Eve can perform attacks to derive the key from data collected by herself and the data transferred between Alice and Bob. The Bloom filter has been used as part of the encoding and perturbation methods in many privacy-preserving applications [29, 30]. Unfortunately, traditional Bloom filter projections cannot remain the order information (without additional process). In other words, the mismatches between the two input key strings and output key strings may be different. In InaudibleKey, we use a special designed Bloom filter data structure considering sequence/order information, which can help project the key into Bloom filter. The purpose of utilizing the adapted Bloom filter is to keep the key distance information while in a non-plaintext format.

The detailed process is presented by the Fig. 4(c). Take a 64-bit key as an example, each single bit position information is conveyed by an addition bit(s) before Bloom filter hash-mapping. Afterwards, two hash functions are used to hash-map each element in the adding-position-bits to the Bloom filtered space. The Bloom filter hash-mapping will only turn the '0' into '1' based on the hash value (calculation) (see [31] for the original rationale). As a result, each '1' in Bloom filter refers to a bit ('0' or '1') at a certain position of the original key 'uniquely'. Most importantly, this adapted Bloom filter data structure can also hold the Jaccard distance between the raw data bits and the projected Bloom filter data bits. That is to say, suppose K_{Alice} and K_{Bob} represent the Bloom filter output of K'_{Alice} and K'_{Bob}, if there are N_{mis} mismatches between K'_{Alice} and K'_{Bob}, then there will be also N_{mis} mismatches between K_{Alice} and K_{Bob}. The proof of this can be referred from [30]. Thus, we can directly use the following information reconciliation approach on K_{Alice} and K_{Bob} because they preserve the similarity information between K'_{Alice} and K'_{Bob}. Note that although the Boom filter is an irreversible one-way function, if the input key's length is too short, Eve can still get the Bloom filter's output, such as through brute-force attack. Therefore, we need to ensure the entropy of the input of the Bloom filter. In InaudibleKey, we concatenate the bits generated from each window to a key string and further divide it into consecutive segments where each segment contains 128 bits. Since Eve has no knowledge of the number and location of the incorrect bits, it is computationally infeasible (2^{128} guesses) to obtain K'_{Alice} and K'_{Bob}. The adapted Bloom filter, which only use hash functions and limit temporary storage (to store the Bloom filter results), will not involve much overhead to our entire mobile system.

4.3 Information Reconciliation

Because of noise, we usually get $K_{Alice} \approx K_{Bob}$ instead of exactly identical keys. The purpose of reconciliation is to correct the mismatches between K_{Alice} and K_{Bob}. In InaudibleKey, we optimise a recent developed CS-based reconciliation method [32] to improve the key agreement rate. To make this paper more self-contained, we first succinctly describe the flow of this method then present our optimisation algorithm.

The key idea of the reconciliation method [32] is *the mismatches between Alice and Bob is much less than that of Alice and Eve (i.e., more sparse)*. Suppose the keys after bloom filter are $K_{Alice} \in R^N$ and $K_{Bob} \in R^N$, respectively. The sampling matrix is $A \in R^{M \cdot N}$ which obeys the restricted isometry property (RIP) [33]. Researches have found a random Bernoulli matrix with equally distributed works well, so we use a random Bernoulli matrix in InaudibleKey. Then Alice and Bob follow the steps below to correct their mismatches.

(1) Firstly, Bob generates a compressed vector $y_{Bob} = A \cdot K_{Bob} \in R^M$ and transmits it to Alice via public channel.
(2) Secondly, after receiving y_{Bob}, Alice calculates the difference between y_{Alice} and y_{Bob}:

$$y_{A,B} = y_{Alice} - y_{Bob} = A(K_{Alice} - K_{Bob}) + e = A\Delta x + e \quad (1)$$

where $y_{Alice} = AK_{Alice}$ and Δx represents the mismatches between K_{Alice} and K_{Bob} and $e \in R^M$ is noise.
(3) Finally, Alice apply ℓ_1 optimization to reconstruct Δx from the compressed data $y_{A,B}$ [33]:

$$\arg\min_{\Delta x} \|\Delta x\|_1 \quad \text{subject to } \|y_{A,B} - A\Delta x\|_2 < \epsilon. \quad (2)$$

where ϵ is used to account for noise. Then, Alice can deduce K_{Bob} by $\bar{K}_{Alice} = K_{Alice} \oplus \Delta x$, and both Alice and Bob agree on the same key $\bar{K}_{Alice} = K_{Bob}$.

When we use the above method, we find that the performance varies a lot. We find out it is because the sampling matrix A is generated randomly. To address this problem, we propose an optimisation problem to improve its performance. According to the theory of compressive sensing [34], the random matrix A must meet the following two conditions.

$$M(A) \leq \frac{C}{\log N}, \quad s \leq \frac{CN}{\log N \cdot \|A\|_2^2} \quad (3)$$

where C is a constant, N is the length of the key (e.g., the number of columns of A), s is the sparsity of the key, and $M(A)$ means the mutual coherence of A which is defined as: $M(A) = \max_{i<j} \frac{|a_i^T a_j|}{\|a_i\|\|a_j\|}$ where a_i and a_j represent the $i-th$ and $j-th$ columns of A respectively. In fact, $M(A)$ represents the maximum value of cross-correlation between columns in A. If A is generated randomly every time, it is hard to ensure it always has good mutual coherence. That is why the performance varies greatly for different A. If we change the second condition above into this form: $\frac{1}{\|A\|_2^2} \geq \frac{s \log N}{CN}$, we can see that if $\|A\|_2^2$ is smaller, it is easier for A to meet the second condition. Therefore, the goal of our proposed optimisation algorithm is to minimise $M(A)$ iteratively.

As shown in Algorithm 1, we start with finding two columns that have minimum coherence from a searching space Ω. Ω includes a large finite set of random matrices. Then in each iteration, we choose a vector from Ω that can minimise the maximum mutual coherence between this vector and those already in A. Finally, when we find N such columns, the iteration terminates and we output the optimised matrix A which has the minimum mutual coherence. This optimisation is conducted offline, so it does not incur any computation overhead. Note that although several other works also optimise projection matrix by minimising mutual coherence or row coherence [35, 36], the problems to be solved are different, i.e., they aim to optimise a projection matrix for a certain dictionary to obtain better recovery signal.

Algorithm 1 Sampling Matrix Optimisation

Objective: Find a matrix A with minimal $M(A)$
Input: Search space Ω, the number of columns of A is N.
Initialisation: traverse Ω to find two column vectors $\hat{a}_1$ and $\hat{a}_2$ such that their coherence is minimal, $\hat{A} = \{\hat{a}_1, \hat{a}_2\}$, $\Omega = \Omega \backslash \{\hat{a}_1, \hat{a}_2\}$, $i = 2$
while $i \leq N$ **do**
  $\tilde{A}_j = \arg\min_{\hat{a}_j \in \Omega} \max_{\hat{a}_k \in \hat{A}} |\hat{a}_j \hat{a}_k|$
  $\hat{A} = \hat{A} \bigcup \{\tilde{a}_j\}$
  $\Omega = \Omega \backslash \{\tilde{a}_j\}$
  i++
end while
Output: optimised matrix $A = [a_1, a_2, \cdots, a_m]$

The optimised reconciliation method presents several advantages. Firstly, Bob only transmits the compressed key instead of the original key. Even if Eve eavesdrops this message, she cannot reconstruct Bob's key K_{Bob} as will be discussed in Sec. 6. So it ensures security. Secondly, unlike some conventional reconciliation methods, this approach does not discard errors during reconciliation process. With the powerful ℓ_1 optimization, this approach can recover more errors than previous reconciliation methods as will be demonstrated in Sec. 5.7. Thus, it can improve key generation rate. Finally, some methods require both Alice and Bob to perform reconciliation to correct the errors between their keys. However, in the CS-based approach, only one device needs to perform reconciliation (e.g., Alice in this example). The users can choose to run ℓ_1 optimization on the power-rich devices to mitigate the computational burden on resource-limited IoT devices. Thus, this approach can improve energy efficiency as will be demonstrated in Sec 5.10.

As discussed in Sec 3, Eve has the power to modify, insert and replay messages. So she can perform two common attacks during reconciliation process: MITM and replay attack. Eve can launch MITM by impersonating as Alice or Bob during key generation process to modify or insert her own messages. To solve this problem, we apply the message authentication code (MAC) method to ensure the integrity and authenticity of the message [5]. Bob includes an additional MAC message with y_{Bob}, so the total message sent to Alice becomes $L_{Bob} = \{y_{Bob}, MAC(K_{Bob}, y_{Bob})\}$. After receiving L_{Bob}, Alice computes K'_{Alice} by Eq. 2 and verifies its identity. If $MAC(K''_{Alice}, y_{Bob}) \neq MAC(K_{Bob}, y_{Bob})$, Alice knows that the message was modified, indicating the presence of Eve. If $MAC(K''_{Alice}, y_{Bob}) = MAC(K_{Bob}, y_{Bob})$, Alice can confirm that this message was indeed originated from Bob. To detect replay attacks, we can adopt some commonly used methods such as nounces, timestamps or tagging each message with a session ID [37].

Although the above methods can prevent MITM and replay attacks, Eve can utilise the eavesdropped y_{Bob} to infer the shared key y_{Bob}. The authors in [32] pointed out two potential vulnerabilities.

Vulnerability 1: Eve can try to recover Bob's key from y_{Bob} by ℓ_1 optimisation directly.

Vulnerability 2: Eve may launch the three types of attacks mentioned in Sec. 3 to obtain her own channel measurement. Then she can impersonate as a legitimate device and perform the same

reconciliation process as Alice by using her channel measurement with the aim of generating the same key.

The authors in [32] have proved that the CS-based reconciliation approach is perfectly effective and secure if the number of rows of A M meets the following condition:

$$P < M < Q, \quad P = S_{\Delta A,B} * log(N/S_{\Delta A,B}), \quad Q = min(S_{Bob}, S_{\Delta B,E})$$

where $S_{\Delta A,B}$ is the sparsity of Δx, S_{Bob} is the sparsity of K_{Bob}, $S_{\Delta B,E}$ is the sparsity of difference between K_{Bob} and K_{Eve}, respectively. The sparsity here means the number of non-zero values in the corresponding vector, i.e., the number of mismatches. As noted in [32], the design for an effective and secure CS-based reconciliation is a problem to find a suitable M, with upper bound Q being the secure threshold, and the lower bound P being the effective threshold. We conduct extensive experiments to find a proper range of M in Sec. 6.

4.4 Privacy Amplification

Although multi-level quantization can generate more bits from one sample, it may also generate some duplicated bits as the example in Fig. 4(b). Directly using such a key will weaken the security of the system. Note that the Bloom filter-based approach in Sec 4.2.2 can prevent reverse engineering attack but cannot improve entropy. To address this problem, we use Karhunen-Loeve Transform (KLT) to decorrelate the bit sequence after reconciliation.

Assume the generated key by Alice after reconciliation is $\bar{K}_{Alice} = (k_1, k_2, \cdots, k_L)^T$, where k_i is the i-th bit and L is the length of the key. The first step in KLT is to calculate the auto-correlation matrix $R = E(KK^T)$. Next, we calculate its eigen value λ_i and eigen vector ϕ_i so that $R\phi = \lambda_i\phi_i \quad (i = 1, 2, \cdots, L)$. Note that R is Hermitian, and its eigenvectors ϕ_i are orthogonal. If we rank eigen values in a descending order, $\lambda_1 > \lambda_2 > \cdots > \lambda_L$, we can construct an unitary matrix Φ which diagonalizes R: $\Phi = [\phi_1, \cdots, \phi_L]$. Φ is called the KLT matrix and can be used to decorrelate the bit sequences K. We choose the largest S eigenvectors to construct Φ, so we can obtain a decorrelated key string by $K''_{Alice} = \Phi^T \bar{K}_{Alice}$. In the same way, Bob can generate a decorrelated key sequence $K''_{Bob} = K''_{Alice}$.

Although reconciliation can improve the reliability of a key generation protocol, it reveals partial information to Eve. Privacy amplification is a common approach to remove the revealed information from the generated secret key sequence. It is usually implemented by the extractor, universal hashing functions, and cryptographic hash functions [38]. In InaudibleKey, we use the commonly used dual universal hash function [39] to generate the final key. Finally, the key can be used by encryption algorithms such as AES-128 to secure their communication.

5 EVALUATION

5.1 Goals, Metrics and Methodology

Experimental Setup. We implement InaudibleKey on Samsung S10 which is equipped with microphone and speaker. The frequency range of the inaudible acoustic signal is 18-22 kHz, and the sampling rate is 48 kHz. The volume of the speaker is set to the maximum, and the corresponding sound pressure level (SPL) of the acoustic signal is 82 dB. Bluetooth Low Energy broadcast is used as the public channel to exchange reconciliation information. These settings can be supported by most modern mobile devices. We conduct extensive experiments with four smartphones, namely, Alice, Bob, Eve, and David (Eve's partner). Same as the state-of-the-art [4, 21], data are collected in four different scenarios: **A**—indoor static, **B**—outdoor static, **C**—indoor mobile, **D**—outdoor mobile. In mobile scenarios, the users can shake Alice and Bob, or carry them and walk around. In static scenarios, Alice and Bob are stationary while some people are moving around. The indoor experiments are conducted in a student laboratory while the outdoor experiments are carried out on a campus road. In each environment, we vary the distance between Alice and Bob from 10 cm to 300 cm to evaluate the impact of distance. Eve and David are located at least 10 cm away from the legitimate devices.

5.2 Parameter Selection

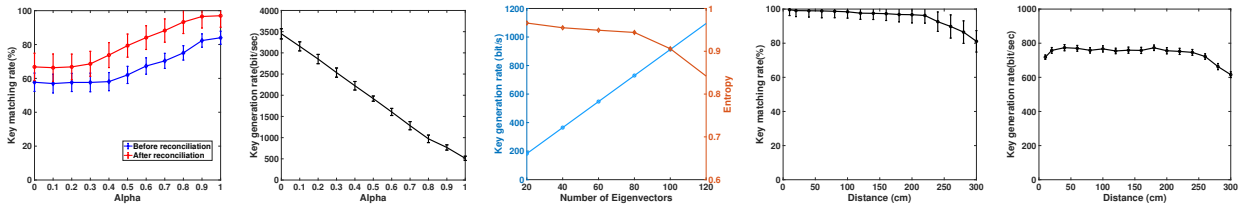
We first evaluate the impact of important parameters in InaudibleKey which include the guard band ratio α in quantization (Sec. 4.2.1) and the number of eigenvectors S in privacy amplification (Sec. 4.4).

The guard band ratio α trades off the key generation rate and key agreement rate, Generally, a larger α means more samples are discarded which improves bit matching rate but decreases the bit generation rate. Fig. 5 shows the impact of α on key generation rate and key matching rate. First, we can see the matching rate increases significantly after information reconciliation. Also, we can see that the key agreement rate rises with the increase of α as more mismatch bits are discarded. In InaudibleKey, we use $\alpha = 0.9$ because it achieves a high key agreement rate while generating sufficient bits. Although a larger α decreases key generation rate, from Fig. 5(b) we can see that it can still achieve a generation rate of 768 bit/sec with $\alpha = 0.9$. The key generation rate of InaudibleKey significantly outperforms the state-of-the-art works: it is 3× faster than FREE [21], 7× faster than TDS [4], 30× faster than Walkie-Talkie [10], 256 × faster than H2B [32], respectively. Therefore, in terms of key generation rate, InaudibleKey is a better option than using radio signal (e.g., TDS [4]), motion sensor signal (e.g., Walkie-Talkie [10], Shake-n-Shack [11]), and biometrics signal (e.g., H2H [40], H2B [32]).

The number of selected eigen vectors S trades off the key generation rate and entropy. If S is larger, InaudibleKey can generate more keys but with lower entropy. If S is smaller, InaudibleKey can generate fewer keys with higher entropy. Fig. 5(c) shows the impact of S on key generation rate and entropy. We can see that if S is less than 80, the improvement of entropy levels off. Therefore, we choose the largest 80 eigenvectors to form the decorrelation matrix.

5.3 Impact of Distance

After determining α, we evaluate the impact of distance between Alice and Bob on system performance. Fig. 5(d) shows that the key agreement rate decreases slightly when the distance between Alice and Bob increases from 10 cm to 220 cm. Then, it starts to drop quickly when the distance further increases from 220 cm to 300 cm. This is because when the distance increases, the audio signal attenuates exponentially due to path loss [41]. From Fig. 5(e), we find that the key generation rate first increases when the distance increases from 10 cm to 20 cm, then becomes stable from 20 cm to 230 cm after which they start to drop rapidly. This is because Line-of-Sight channel dominates the signal when two devices are very close



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

Figure 5: Impact of α, S and distance.

and hence there is not much randomness to use. However, when the communication distance is too large, more environmental noise is involved in the received signal and the signal-to-noise ratio (SNR) becomes low, which leads to more discrepancies. From Fig. 5(d) and Fig. 5(e), we find that [20 cm, 220 cm] is a reasonable pairing range to achieve both high agreement rate and bit generation rate. InaudibleKey extends the pairing distance by 3.2 times compared to FREE [21], and 44 times compared to TDS [4].

Researchers have revealed that people's social distance varies from 1.2 m to 3.7 m [42]. Therefore, InaudibleKey can meet the pairing requirement of mobile devices in most cases. If the distance of two users is larger than pairing range (say >3m), user can either walk a few steps closer to the target or use a shorter key for temporary association, depending on the application requirement and user's preference. In particular, nowadays people should keep 1.5–2 m social distance due to pandemic. Our system can help users establish a secure communication channel without close contact, and thus help stop the spread of COVID-19.

5.4 Impact of Different Environments

In this experiment, we evaluate the impact of different environments by using the data in the range of [20 cm, 220 cm]. Fig. 6(a) illustrates the key generation rate and key agreement rate in different scenarios. Intuitively, the key agreement rate of outdoor environment (i.e, B and D) is slightly lower than that of indoor environment (i.e., A and C). This is because there are less multi-path effect and more environmental noise in outdoor environment [6]. In terms of generation rate, we can see that the mobile scenarios (i.e., C and D) can generate more bits in comparison with static scenarios (i.e., A and B). This is because the mobile scenarios can generate more channel diversity and randomness.

5.5 Impact of Background Noise

While our analysis above show that InaudibleKey consistently achieve high agreement rate, the experiments are conducted on campus only. The real-world environment is more complex and may contain various kinds of noise. We now study the degradation in matching rate with increasing background noise. We manually add 18-22 kHz random Gaussian noise with different intensities (30, 50 and 70 dB). From Fig. 6(b), we can see that the key agreement rate only drops slightly when the noise level is 30 dB. However, when we further increase the noise level, the key agreement rate drops significantly. Actually, in the range of 18-22 kHz, there are little environment noise in the normal office and street environments, and such noise usually happens in factory and metro station [20].

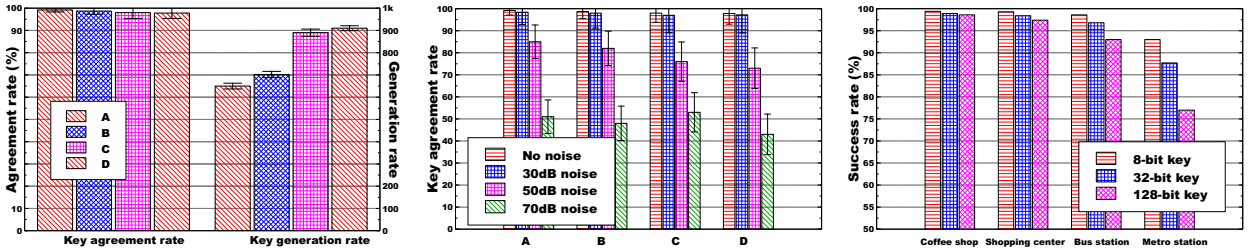
To verify this, we conducted another experiment in four common environments: coffee shop, shopping centre, bus station and metro station. The distance between Alice and Bob is about 1 m, and we collect 30 minutes data for each environment. All measurements are made between 9 AM and 6 PM. In this experiment, we use success rate (the probability of generating the same key) instead of matching rate as metric because we want to know how many trials the users need to successfully pair two devices in real environments. Fig. 6(c) shows the success rate of each environment. We can see that InaudibleKey can achieve over 95% success rate in coffee shop and shopping centre. The success rate in bus station drops slightly, but it still can achieve over 90% success rate. We notice that the success rate in metro station drops remarkably. This is because the noise level of subway stations can be up to over 100 dB according to prior studies [43, 44]. More importantly, it has more noise in the inaudible frequency range. However, if we use 8-bit key, the success rate is still above 92%. The results suggest that in noisy environment, users can use a short key to improve success rate.

5.6 Impact of Device Diversity

So far, we assume both Alice and Bob use the same model of smartphones (i.e., Samsung S10). Now we evaluate the performance of InaudibleKey when Alice and Bob are different types of devices. Fig. 7 plots the CFRs of different types of mobile devices, we can see that when Alice and Bob are using Samsung S10, their CFRs are very close to each other. While when Bob is using HTC smartphone and HUAWEI watch, it involves more differences, but the CFR pattern over the frequency band is still very similar. Tab. 1 shows the key agreement rate using different devices. It is intuitive that InaudibleKey achieves the highest success rate when Alice and Bob are the same type of devices. The success rate drops slightly when Alice and Bob are different types of devices. This is because different types of microphone and speaker produce different impact in the transmitted and recorded acoustic signals. In particularly, the matching rate of Arduino with other devices are the lowest because we use a low-price microphone and speaker module as will be discussed in Sec. 5.10.

5.7 Comparison of Reconciliation Methods

To demonstrate the advantage of our optimisation algorithm, we compare it with the original CS-based reconciliation method and other methods. In the literature, two commonly used reconciliation techniques are error-correcting code (ECC) [23, 45] and interactive method [5, 10, 46, 47]. In this paper, we use Reed-Solomon code RS(15,7) and the method in [46] as benchmark. We calculate the key



(a) Impact of different environments (b) Impact of background noise (c) Performance in real environments

Figure 6: Impact of environment and noise.

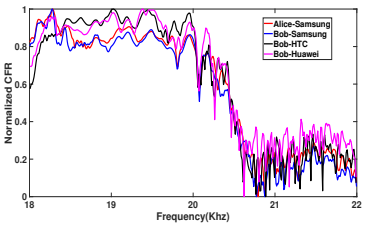


Figure 7: CFR of different devices

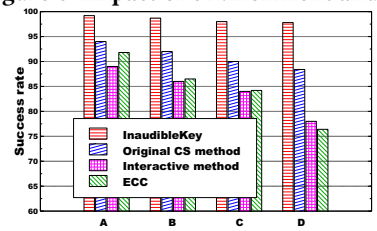


Figure 8: Different reconciliation methods.

	Samsung	HTC	Huawei	Arduino
Samsung	**99.2%**	94.5%	94.1%	89.3%
HTC	94.5%	**98.7%**	92.4%	87.7%
Huawei	94.1%	92.4%	**98.8%**	86.4%
Arduino	89.3%	87.7%	86.4%	**95.6%**

Table 1: Success rate of different pairs.

agreement rate of each method and plot their results in Fig. 8. The result of original CS-based method is obtained by averaging the results of randomly generating sampling matrix 30 times. As can be seen from Fig. 8, InaudibleKey outperforms the original CS-based method, interactive method and ECC and consistently achieves the highest agreement rate in all the environments.

5.8 Comparison with state-of-the-arts

In this subsection, we compare InaudibleKey with several representative key agreement approaches for mobile networks. These methods include KEEP [1], ASBG [3], TDS [4], Radio-telepathy [5], CGC [6] and FREE [21]. To achieve a fair comparison, we fine-tune the parameters of other methods to make sure they achieve the best performance. Specifically, for FREE, the distance between Alice and Bob is set to 80cm, and the block size is 30. For ASBG, KEEP and CGC, α and fragment size are set as 0.35 and 50, respectively. For TDS, the block size β is 5. The distance between Alice and Bob is set to 4 cm as suggested by their authors. We compare the key agreement rate, key generation rate, entropy, and reconciliation counts of different methods.

Fig. 9 shows the performances of different approaches. From Fig. 9(a), we can see that the key agreement rate of InaudibleKey is higher than other approaches except for TDS. In this experiment, TDS can achieve the highest agreement rate because of the short distance (4 cm). However, such short distance is unrealistic in practice. From Fig. 9(b), we can see that the key generation rate of InaudibleKey is significantly higher than previous works. To be specific, the key generation rate of InaudibleKey is 3× faster than FREE [21], 7× faster than TDS [4] on average. There are a number of reasons for the improvement. First, the sampling rate of the audio signal (i.e., 48kHz) is significantly higher than the radio channel probing. Second, the channel frequency response can provide more channel information compared to channel tap used in FREE [21]. Finally, the optimised CS-based reconciliation methods can recover more mismatches as demonstrated in the last subsection.

Fig. 9(c) shows the entropy of extracted keys. We find that by using KLT to decorrelate the bit sequences, InaudibleKey achieves higher entropy than other methods. Fig. 9(d) plots the information reconciliation counts of different methods. We can see that InaudibleKey requires the minimum information reconciliation counts. To successfully generate the same key, InaudibleKey only requires Alice and Bob to exchange reconciliation messages 1.6 times on average. In comparison, TDS requires 4 pass checks [4], and FREE needs to exchange 25 reconciliation information messages on average [21]. In other words, InaudibleKey reduces information reconciliation counts by 2.5-16 times.

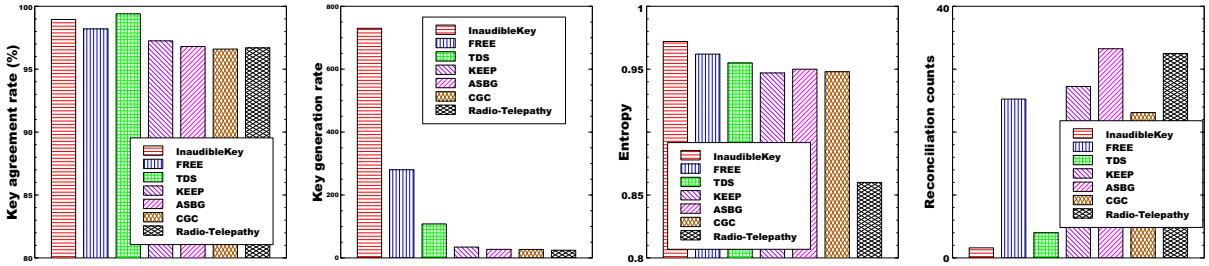
The results show that InaudibleKey improves the key generation rate, the entropy, and reduces reconciliation counts significantly compared to the state-of-the-arts.

5.9 Randomness of Key

To evaluate the randomness of the extracted keys, we apply the commonly used NIST suite of statistical tests [48]. The result of NIST statistical test are p-values of different test processes which indicate whether the key is random or not. If p-value is greater than 1%, then the key is considered to be random. From the results in Tab. 2, we find that the p-values are all larger than 1%, which suggests the extracted keys have good quality in randomness.

Table 2: Results of NIST test.

NIST TEST	p-value
Serial	0.553
FFT Test	0.179
Longest Run	0.353
Monobit Frequency	0.742
Linear Complexity	0.705
Block Frequency	0.178
Cumulative Sums	0.741
Approximate Entropy	0.885
Non Overlapping Template	0.532



(a) Key agreement rate (b) Key generation rate (c) Entropy (d) Reconciliation counts

Figure 9: Comparison with state-of-the-arts.

Table 3: System overhead.

	Samsung S10			Arduino		
	InaudibleKey	RSA	ECDHE-RSA	InaudibleKey	RSA	ECDHE-RSA
Processing Time (ms)	124	361	347	891	4,196	5,481
Energy Consumption (mJ)	108	391	354	1,107	1,706	2,196

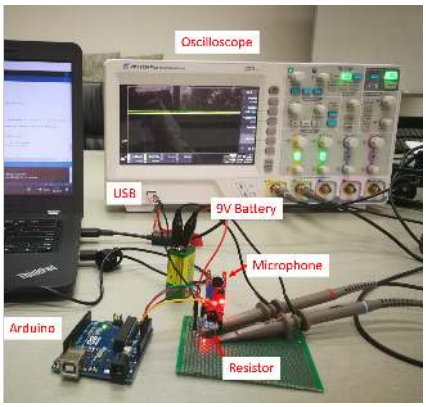


Figure 10: Experimental setup of energy consumption.

5.10 System Implementation

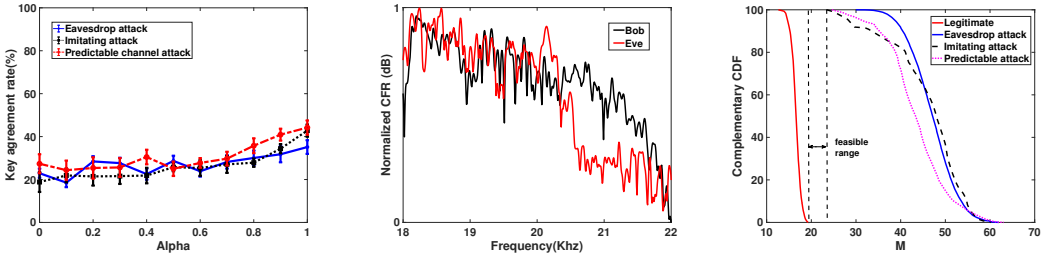
To validate the feasibility of InaudibleKey on various IoT devices, we implement the prototype of InaudibleKey on Samsung S10 smartphone and Arduino Uno board.

The CPU of Samsung S10 is an Snapdragon at 2.84 GHz and the operating system is Android 9.0. It is equipped with a stereo speaker and two dedicated microphone with active noise cancellation function. Only the bottom microphone is used because it is close to the speaker. The system is implemented in Java and the MAC algorithm described in Section 4.3 is implemented based on SHA256 (HMAC-SHA256). In InaudibleKey, we save the transmitted OFDM signal as a Waveform Audio (WAV) file with a format of 16-bit Pulse Coded Modulation (PCM), which will be played by speaker. To reduce the expected response time, we implement InaudibleKey in multiple threads. Two threads are created after InaudibleKey is launched. One of the threads is responsible for transmitting WAV file. After transmitting, the smartphone will transit into listening mode and another thread which records audio signal from another phone will be created. In reconciliation, InaudibleKey uses ℓ_1-*Homotopy* [49] which is an efficient implementation of ℓ_1 optimization algorithm. The complexity of ℓ_1-*Homotopy* is $O(k^3 + kmn)$, where k is the sparsity of the solution, m and n are the size of sampling matrix A which is 23 and 128, respectively.

Firstly, we compare InaudibleKey with public key cryptography and Diffie-Hellman key exchange protocol. For public key cryptography, we use the commonly used RSA as benchmark. Alice can use RSA to encrypt a 128 bits key which can be decrypted by Bob. Then, Alice and Bob can use AES-128 to secure their communication. In this experiment, we use 2048 bits key for RSA which is recommended by NIST [50]. Traditional Diffie-Hellman protocol is susceptible to MITM attack, and is rarely used in practice. Therefore, we use the commonly used Elliptic Curve Diffie Hellman Ephemeral with RSA signature (ECDHE_RSA), which is used in Transport Layer Security (TLS). We implement these algorithms on Samsung S10 and calculate their processing time and energy consumption. The implementation of these cryptographic algorithms are based on the Chilkat library [3]. The computation time is obtained from the console of the development environment (Android studio) and averaged by the results from 30 tests. The energy consumption of smartphone is calculated by reading the voltage and current level of the battery which can be obtained by Android API [4]. As the results in Table 3, we can see that RSA requires 361 ms to finish a round of encryption and decryption with a 2048 bits key. It takes about 347 ms for ECDHE_RSA to generate a 128 bits key. However, InaudibleKey only requires 124 ms to generate a 128 bits key. Therefore, InaudibleKey is superior to public key cryptography and D-H protocol in key distribution on mobile devices.

Secondly, to verify the feasibility of InaudibleKey on resource-limited IoT devices, we implement our system on Arduino Uno board. Compared to the powerful CPU in Samsung S10, the microcontroller of Arduino is ATmega328 which only has 32 K flash memory and 2 K SRAM. There is no default speaker and microphone on Arduino board, so we connect additional speaker module and microphone module to it. To measure the energy consumption on Arduino, we connect the output of a 9V battery [5] to digital oscilloscope. The details of the experimental setup is shown in Fig. 10. The voltage over the resistor is stored in USB and used

[3] http://www.chilkatsoft.com/
[4] https://developer.android.com/reference/android/os/BatteryManager.html
[5] The operating voltage of Atmega328p is 5V, but the input voltage of the Arduino board is 6V to 12V.



(a) Agreement rate of different attacks (b) Bob vs predictable channel attacker (c) Feasible range of parameter M

Figure 11: Security analysis.

to calculate the energy consumption of the board. The processing time and energy consumption is shown in Table 3. We can see that although the system overhead of InaudibleKey is much higher than that on smartphone, it is still much more efficient than RSA and ECDHE-RSA. Moreover, the computation-intensive part of InaudibleKey—reconciliation— can be performed on the power-rich device (if one of the devices is powerful).

We now analyse the impact of energy consumption on IoT devices. The battery capacity of the Samsung S10 is 3,400 mAh (42.8 kJ). So, the energy cost of InaudibleKey amounts to $0.3e^{-5}$ of the total energy supply. If we assume the smartphone with a targeted lifespan of one day which results in an energy budget of 1.75 kJ per hour. Then, with only 1% of the battery budget (17.5 J), InaudibleKey is able to run approximately 175 times per hour, i.e., InaudibleKey can continuously run every 20 seconds. In the same way, we can estimate that with 9V battery (500 mAh) and 1% of the battery budget, InaudibleKey is able to run every 22 minutes on Arduino Uno board, i.e., it can run about 3 times per hour. These results demonstrate that InaudibleKey incurs a low system overhead and is more efficient than public key scheme.

6 SECURITY ANALYSIS

6.1 Against Vulnerability 1

In Vulnerability 1, Eve can try to reconstruct the keys from y_{Bob} directly using ℓ_1 optimization. As discussed above, the key generated by InaudibleKey has high entropy, which means almost half of the keys are bit '1's. Fig. 6(a) shows that the initial agreement rate of Alice and Bob is about 84% when $\alpha = 0.9$. Assume we use 128-bit key, we have $P = 128 \times 0.84 \times log(128/(128 \times 0.84)) \approx 19$ and $Q = 128 \times 50\% = 64$ according to [32]. Theoretically, the range of M can be [20,63] because $P < M < Q$. Practically, we can choose $M \in [23, 50]$ to guarantee security against an adversary and the availability of the same key.

6.2 Against Vulnerability 2

Eve can perform the following three types of attacks to generate a key K_{Eve} that is close to K_{Bob} hoping that she can recover K_{Bob} with the eavesdropped y_{Bob}.

Against Eavesdropping Attack. In this attack, Eve can eavesdrop all the communication traffic in the public channel. Since Eve is located out of the safe distance (>10 cm), she will obtain a totally different channel response, as discussed in Sec. 2. From Fig. 11(a), we can see that the agreement rate of eavesdropping attack is about 20-35%. Therefore, if Eve is out of the safe distance, she cannot guess the same key due to the different multipath fading channel.

Against Imitating Attack. In this attack, Eve can observe how Alice and Bob generate keys. Then, after Alice and Bob leave the site, Eve will ask her partner David to imitate the motion of Alice and Bob to generate the same key. Previous studies have shown that simply imitating the user's shaking or walking motions cannot generate the same key for accelerometer-based authentication systems [10, 11, 51]. Similarly, Fig. 11(a) shows that an imitating attack can achieve a higher agreement rate when α increases. But eventually, it can at most achieve 42% agreement rate. More importantly, Eve does not know which bit is correct because of the time-varying nature of channels.

Against Predictable Channel Attack. Predictable channel attack is a simple but effective attack to compromise a key agreement protocol, especially for RSSI-based approaches [3, 6]. In this attack, Eve can intentionally block and unblock the Line-of-Sight (LOS) between Alice and Bob to generate predictable channel measurements. We evaluate this attack by setting up Alice and Bob 100 cm away with LOS and ask a person to walk between Alice and Bob intentionally. Then after the key generation, we replace Alice and Bob with Eve and David and ask the same person to repeat the process. Then we compare Eve's key with Bob's key to see if Eve can generate the same key. From Fig. 11(a), we can see that a predictable channel attack can achieve the highest matching rate among these three types of attack. But still, it can at most reach 43% matching rate. Fig. 11(b) plots the CFR of Bob and Eve when the same person blocks the LOS signal. We can see that although the channel responses of Bob and Eve are similar in some frequencies, there is still a large portion of the difference in other frequencies due to time-varying channels and hardware difference. Particularly, we noticed that Eve is capable of producing similar channel responses in the lower frequency range but not the higher frequency range. There are two reasons. First, the microphone actually works as a low-pass filter with a 22 kHz cutoff frequency [52]. So in the higher frequency range, the acoustic signal will be attenuated slightly which results in more mismatches. Additionally, Zhou et al. [20] found that different speakers' performances are much more diversified at a higher frequency range. If the attacker leverages more sophisticated hardware, it is possible that they increase their attacking ability. But it is an open question that requires further investigation.

Although imitating attack and predictable channel attack can achieve approximately 43% matching rate, the probability of deducing the same 128-bit key is extremely low, i.e., $0.43^{128} = 1.21e^{-47}$. The matching rate of Eve can be further reduced by setting a higher threshold in quantization or turning down the volume of speaker. Considering the mathing rate of Eve, a 225-bit key of our system is equivalent to a 128-bit AES symmetric key, and it takes about 0.3 s to generate such a key based on the result in Sec. 5.3.

Fig. 11(c) shows the distribution of P and Q in our dataset. We can see that there is a feasible range to use. In other words, if M lies in the feasible range, then InaudibleKey is resilient to the three types of attacks above. Previous studies also found that if the same sampling matrix A is used repeatedly, both y_{Bob} and K_{Bob} could be conditionally accessed [53]. We can easily solve this problem by updating A after each successful key generation. Although A needs to be pre-stored, it is public information instead of a secret that is only known by Alice and Bob. InaudibleKey does not realise literal authentication but rely on the user to authenticate the other device. This is practical because if Eve wants to perform impersonation attack, she should be close to Alice or Bob. Her suspicious actions can be easily spotted by Alice or Bob.

7 RELATED WORK

Proximity-based approaches. The proximity-based approaches pair two devices based on the observation that two devices in physical proximity can measure similar physical information. Researchers have proposed many different systems by exploring various location-sensitive features such as RSSI [7, 8, 54], CSI [4], audio [55] and illumination [56]. However, these approaches suffer from a common problem: the distance between two legitimate devices should be very close, e.g., 1.25 cm in Proximate [8] and 5cm in TDS [4].

Channel reciprocity-based approaches. Physical layer key generation is a hot research filed over the past decade. Researchers have studied key agreement for different wireless technologies such as ZigBee [3], Wi-Fi [4, 5]. Among these approaches, RSSI-based key generation methods suffer from predictable channel attack and low bit generation rate. Although CSI-based key generation methods can improve bit generation rate, most systems rely on customised hardware to obtain CSI information. Recently, researchers also use unique body channel to pair two mobile devices [17, 57]. However, these methods require specialised sensors such as electrode [17] and Electromyogram sensor [57].

Acoustic signal-based approaches. Recently, the acoustic signal is also exploited to pair mobile devices [15, 16, 21, 22, 55]. Proximity-based schemes such as [16, 55] are not feasible due to constraint of social distance. Two recent works [21, 22] are closely related to our system. FREE [21] used channel tap and the authors of [22] used sound pressure as channel characteristics. However, these metrics can only provide a coarse estimation of acoustic channel. In comparison, we modulate the audio signal using OFDM technology to obtain fine-grained channel estimation and propose an optimisation algorithm to improve the performance of reconciliation. This is why we can achieve much higher generation rate.

8 CONCLUSION

In this paper, we propose a novel key generation system for mobile devices via inaudible acoustic signal. Extensive evaluation results show that InaudibleKey outperforms the state-of-the-arts significantly. To demonstrate the feasibility, we implement InaudibleKey on both powerful and resource-limited IoT devices. We also verify the security of InaudibleKey against malicious attacks. The results in this paper show that InaudibleKey is a fast, practical and efficient key generation protocol for mobile devices that can work in various environments. More importantly, it allows users to pair two mobile devices without breaking social distance restrictions.

ACKNOWLEDGMENTS

This work is supported by the APRC grant (Project No. 9610485) and the Start-up grant (Project No. 7200642) from City University of Hong Kong.

REFERENCES

[1] Wei Xi, Xiang-Yang Li, Chen Qian, Jinsong Han, Shaojie Tang, Jizhong Zhao, and Kun Zhao. Keep: Fast secret key extraction protocol for d2d communication. In *IWQoS*, pages 350–359. IEEE, 2014.
[2] Weitao Xu, Junqing Zhang, Shunqi Huang, Chengwen Luo, and Wei Li. Key generation for internet of things: A contemporary survey. *ACM Computing Surveys (CSUR)*, 54(1):1–37, 2021.
[3] Suman Jana, Sriram Nandha Premnath, Mike Clark, Sneha K Kasera, Neal Patwari, and Srikanth V Krishnamurthy. On the effectiveness of secret key extraction from wireless signal strength in real environments. In *Mobicom*, pages 321–332. ACM, 2009.
[4] Wei Xi, Chen Qian, Jinsong Han, Kun Zhao, Sheng Zhong, Xiang-Yang Li, and Jizhong Zhao. Instant and robust authentication and key agreement among mobile devices. In *CCS*, pages 616–627. ACM, 2016.
[5] Suhas Mathur, Wade Trappe, Narayan Mandayam, Chunxuan Ye, and Alex Reznik. Radio-telepathy: extracting a secret key from an unauthenticated wireless channel. In *Mobicom*, pages 128–139. ACM, 2008.
[6] Hongbo Liu, Yang Wang, Jie Yang, and Yingying Chen. Fast and practical secret key extraction by exploiting channel response. In *INFOCOM*, pages 3048–3056. IEEE, 2013.
[7] Alex Varshavsky, Adin Scannell, Anthony LaMarca, and Eyal De Lara. Amigo: Proximity-based authentication of mobile devices. In *Ubicomp*, pages 253–270. Springer, 2007.
[8] Suhas Mathur, Robert Miller, Alexander Varshavsky, Wade Trappe, and Narayan Mandayam. Proximate: proximity-based secure pairing using ambient wireless signals. In *Mobisys*, pages 211–224. ACM, 2011.
[9] Nikolaos Karapanos, Claudio Marforio, Claudio Soriente, and Srdjan Capkun. Sound-proof: usable two-factor authentication based on ambient sound. In *24th USENIX Security Symposium*, pages 483–498, 2015.
[10] Weitao Xu, Girish Revadigar, Chengwen Luo, Neil Bergmann, and Wen Hu. Walkie-talkie: Motion-assisted automatic key generation for secure on-body device communication. In *IPSN*, pages 1–12. IEEE, 2016.
[11] Yiran Shen, Fengyuan Yang, Bowen Du, Weitao Xu, Chengwen Luo, and Hongkai Wen. Shake-n-shack: Enabling secure data exchange between smart wearables via handshakes. In *PerCom*, pages 1–10. IEEE, 2018.
[12] Qian Wang, Hai Su, Kui Ren, and Kwangjo Kim. Fast and scalable secret key generation exploiting channel phase randomness in wireless networks. In *INFOCOM*, pages 1422–1430. IEEE, 2011.
[13] Daniel Halperin, Wenjun Hu, Anmol Sheth, and David Wetherall. Tool release: Gathering 802.11 n traces with channel state information. *ACM SIGCOMM Computer Communication Review*, 41(1):53–53, 2011.
[14] Matthias Schulz, Jakob Link, Francesco Gringoli, and Matthias Hollick. Shadow wi-fi: Teaching smartphones to transmit raw signals and to extract channel state information to implement practical covert channels over wi-fi. In *Mobisys*, pages 256–268, 2018.
[15] Jun Han, Albert Jin Chung, Manal Kumar Sinha, Madhumitha Harishankar, Shijia Pan, Hae Young Noh, Pei Zhang, and Patrick Tague. Do you feel what i hear? enabling autonomous iot device pairing using different sensor types. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 836–852. IEEE, 2018.
[16] Pengjin Xie, Jingchao Feng, Zhichao Cao, and Jiliang Wang. Genewave: Fast authentication and key agreement on commodity mobile devices. *IEEE/ACM Transactions on Networking (TON)*, 26(4):1688–1700, 2018.

[17] Marc Roeschlin, Ivan Martinovic, and Kasper Bonne Rasmussen. Device pairing at the touch of an electrode. In *NDSS*, volume 18, pages 18–21, 2018.

[18] Wei Wang, Lin Yang, and Qian Zhang. Touch-and-guard: secure pairing through hand resonance. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 670–681, Heidelberg Germany, September 2016. ACM.

[19] Rajalakshmi Nandakumar, Vikram Iyer, Desney Tan, and Shyamnath Gollakota. Fingerio: Using active sonar for fine-grained finger tracking. In *CHI*, pages 1515–1525. ACM, 2016.

[20] Zhe Zhou, Wenrui Diao, Xiangyu Liu, and Kehuan Zhang. Acoustic fingerprinting revisited: Generate stable device id stealthily with inaudible sound. In *CCS*, pages 429–440. ACM, 2014.

[21] Youjing Lu, Fan Wu, Shaojie Tang, Linghe Kong, and Guihai Chen. Free: A fast and robust key extraction mechanism via inaudible acoustic signal. In *Mobihoc*, pages 311–320. ACM, 2019.

[22] Dania Qara Bala and Bhaskaran Raman. Phy-based key agreement scheme using audio networking. In *2020 International Conference on COMmunication Systems & NETworkS (COMSNETS)*, pages 129–136. IEEE, 2020.

[23] Hongbo Liu, Jie Yang, Yan Wang, and Yingying Chen. Collaborative secret key extraction leveraging received signal strength in mobile wireless networks. In *INFOCOM*, pages 927–935. IEEE, 2012.

[24] David Tse and Pramod Viswanath. *Fundamentals of wireless communication*. Cambridge university press, 2005.

[25] Matthew Edman, Aggelos Kiayias, and Bülent Yener. On passive inference attacks against physical-layer key extraction? In *Proceedings of the Fourth European Workshop on System Security*, page 8. ACM, 2011.

[26] John Paul Walters, Zhengqiang Liang, Weisong Shi, and Vipin Chaudhary. Wireless sensor network security: A survey. *Security in distributed, grid, mobile, and pervasive computing*, 1:367, 2007.

[27] Theodore S Rappaport et al. *Wireless communications: principles and practice*, volume 2. prentice hall PTR New Jersey, 1996.

[28] Kuan Zhang, Patricia Werner, Ming Sun, F Xavier Pi-Sunyer, and Carol N Boozer. Measurement of human daily physical activity. *Obesity research*, 11(1):33–40, 2003.

[29] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, pages 1054–1067, 2014.

[30] Wanli Xue, Dinusha Vatsalan, Wen Hu, and Aruna Seneviratne. Sequence data matching and beyond: New privacy-preserving primitives based on bloom filters. *IEEE Transactions on Information Forensics and Security*, 2020.

[31] Burton H Bloom. Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM*, 13(7):422–426, 1970.

[32] Qi Lin, Weitao Xu, Jun Liu, Abdelwahed Khamis, Wen Hu, Mahbub Hassan, and Aruna Seneviratne. H2b: heartbeat-based secret key generation using piezo vibration sensors. In *IPSN*, pages 265–276. ACM, 2019.

[33] D.L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, pages 1289–1306, 2006.

[34] E.J. Candes and Emmanuel J. The restricted isometry property and its implications for compressed sensing. *Comptes Rendus Mathematique.*, pages 589–592, 2008.

[35] Yiran Shen, Wen Hu, Mingrui Yang, Bo Wei, Simon Lucey, and Chun Tung Chou. Face recognition on smartphones via optimised sparse representation classification. In *IPSN*, pages 237–248. IEEE Press, 2014.

[36] Michael Elad. Optimized projections for compressed sensing. *IEEE Transactions on Signal Processing*, 55(12):5695–5702, 2007.

[37] Sreekanth Malladi, Jim Alves-Foss, and Robert B Heckendorn. On preventing replay attacks on security protocols. Technical report, IDAHO UNIV MOSCOW DEPT OF COMPUTER SCIENCE, 2002.

[38] Junqing Zhang, Trung Q Duong, Alan Marshall, and Roger Woods. Key generation from wireless channels: A review. *Ieee access*, 4:614–626, 2016.

[39] Masahito Hayashi and Toyohiro Tsurumaru. More efficient privacy amplification with less random seeds via dual universal hash function. *IEEE Transactions on Information Theory*, 62(4):2213–2232, 2016.

[40] Masoud Rostami, Ari Juels, and Farinaz Koushanfar. Heart-to-heart (h2h): authentication for implanted medical devices. In *CCS*, pages 1099–1112. ACM, 2013.

[41] Attenuation of sound waves. https://www.nde-ed.org/EducationResources/CommunityCollege/Ultrasonics/Physics/attenuation.htm, 2019.

[42] Edward T et al. Hall. Proxemics [and comments and replies]. *Current anthropology*, 9(2/3):83–108, 1968.

[43] Donguk Lee, Gibbeum Kim, and Woojae Han. Analysis of subway interior noise at peak commuter time. *Journal of audiology & otology*, 21(2):61, 2017.

[44] Robyn RM Gershon, Richard Neitzel, Marissa A Barrera, and Muhammad Akram. Pilot survey of subway and bus stop noise levels. *Journal of Urban Health*, 83(5):802, 2006.

[45] Weitao Xu, Chitra Javali, Girish Revadigar, Chengwen Luo, Neil Bergmann, and Wen Hu. Gait-key: A gait-based shared secret key generation protocol for wearable devices. *ACM Transactions on Sensor Networks (TOSN)*, 13(1):6, 2017.

[46] Kai Zeng, Daniel Wu, An Chan, and Prasant Mohapatra. Exploiting multiple-antenna diversity for shared secret key generation in wireless networks. In *INFOCOM*, pages 1–9. IEEE, 2010.

[47] Lili Meng, Jie Liang, Upul Samarawickrama, Yao Zhao, Huihui Bai, and André Kaup. Multiple description coding with randomly and uniformly offset quantizers. *IEEE Transactions on Image Processing*, 23(2):582–595, 2014.

[48] Andrew Rukhin, Juan Soto, James Nechvatal, Miles Smid, and Elaine Barker. A statistical test suite for random and pseudorandom number generators for cryptographic applications. Technical report, Booz-Allen and Hamilton Inc Mclean Va, 2001.

[49] D.L. Donoho and Y. Tsaig. Fast Solution of ℓ_1 -Norm Minimization Problems When the Solution May Be Sparse. *IEEE Transactions on Information Theory*, 54(11):4789–4812, 2008.

[50] Recommendation for key management. https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57Pt3r1.pdf.

[51] Rene Mayrhofer and Hans Gellersen. Shake well before use: Intuitive and secure pairing of mobile devices. *IEEE Transactions on Mobile Computing*, 8(6):792–806, 2009.

[52] Yitao He, Junyu Bian, Xinyu Tong, Zihui Qian, Wei Zhu, Xiaohua Tian, and Xinbing Wang. Canceling inaudible voice commands against voice control systems. In *Mobicom*, pages 1–15, 2019.

[53] Zuyuan Yang, Wei Yan, and Yong Xiang. On the security of compressed sensing-based signal cryptosystem. *IEEE Transactions on Emerging Topics in Computing*, 3(3):363–371, 2015.

[54] Jiansong Zhang, Zeyu Wang, Zhice Yang, and Qian Zhang. Proximity based iot device authentication. In *INFOCOM*, pages 1–9. IEEE, 2017.

[55] Dominik Schürmann and Stephan Sigg. Secure communication based on ambient audio. *IEEE Transactions on Mobile Computing*, 12(2):358–370, 2011.

[56] Markus Miettinen, N Asokan, Thien Duc Nguyen, Ahmad-Reza Sadeghi, and Majid Sobhani. Context-based zero-interaction pairing and key evolution for advanced personal devices. In *CCS*, pages 880–891. ACM, 2014.

[57] Lin Yang, Wei Wang, and Qian Zhang. Secret from muscle: Enabling secure pairing with electromyography. In *Sensys*, pages 28–41. ACM, 2016.