Wireless Netw (2019) 25:733–751
https://doi.org/10.1007/s11276-017-1588-9

PSP: proximity-based secure pairing of mobile devices using WiFi signals

Weirong Cui[1] · Chenglie Du[1] · Jinchao Chen[1]

Published online: 26 September 2017

Abstract Wireless device-to-device (D2D) communication, which enables direct communication between co-located devices without Internet access, is becoming common. Simultaneously, security issues have become technical barriers to D2D communication due to its "open-air" nature and lack of centralized control. Automatically establishing the secure association between wireless devices that do not share a prior trust remains an open and challenging problem. Recent work has proposed to extract shared keys from the similar ambient radio signals of two co-located wireless devices. Using such methods, information reconciliation based on error-correcting techniques is implemented to make two co-located devices extract the same bitstreams as the shared keys from their similar ambient radio environment. However, due to the bounded capability of the error-correcting code, existing methods can only work effectively in a very short distance range. In this paper, we propose a novel solution, called proximity-based secure pairing (PSP), which allows two wireless devices in physical proximity to automatically authenticate each other and obtain shared keys according to the channel state information of the WiFi signals. In contrast to existing methods, PSP is built on private set intersection computation rather than information reconciliation, which makes it effective over a wider distance range while ensuring security and efficiency. We provide a thorough security analysis and performance evaluation of PSP and demonstrate its advantages in terms of security, efficiency and usability over state-of-the-art methods.

Keywords Secure pairing · Key agreement · WiFi · Channel state information

1 Introduction

Background Wireless device-to-device communication (D2D), which enables direct communication between nearby mobiles, has become common in recent years. For example, in mobile social networks (MSNs), people can use a smartphone to directly exchange various types of information such as texts, pictures, and videos [1, 2]. In wireless sensor networks (WSNs), sensors can directly communicate with each other to collaboratively sense the physical world [3]. Authentication and key agreement are fundamental requirements in ensuring the security of D2D communication. However, compared to traditional network communication, satisfying such requirements for D2D is more challenging due to the broadcast nature of the wireless medium and the lack of centralized management.

The problem that we want to address in this paper is how to securely pair two co-located wireless devices that do not share a prior trust relationship in the D2D communication, also called *proximity-based secure pairing*. More specifically, proximity-based secure pairing is actually the security handshake process prior to data transfer in D2D communication. In this process, two devices located in physical proximity should be able to authenticate each other and negotiate the shared communication keys.

Current research To achieve proximity-based secure pairing, the most straightforward method is to take advantage of human involvement. For example, in the seeing-is-

✉ Weirong Cui
sealrong@163.com

[1] School of Computer Science and Engineering, Northwestern Polytechnical University, Xi'an 710072, People's Republic of China

believing system proposed by McCune et al. [4], the camera of a mobile device is incorporated to capture the 2D-barcode displayed on a second device to verify the device's identity. In the loud and clear system proposed by Goodrich et al. [5], a text message displayed on one device is read aloud by a user and recorded again via speech recognition by a second device. As a further example, Mayrhofer et al. present an authentication mechanism based on accelerometer readings of the simultaneous shaking processes of devices [6]. However, such methods have two major limitations. First, with the global trend toward miniaturization and increased variety of device form factors, the devices may not have a common set of hardware components required for establishing a human-supported secure association. For instance, the devices may or may not have a screen, buttons, LEDs, accelerometers, RFIDs and NFC chips [7]. Second, in certain application scenarios, such as WSNs, devices have to spontaneously establish a secure link without human involvement.

In recent years, a new idea for implementing proximity-based secure paring without human involvement has been proposed in academia. This idea allows devices located in close physical proximity to derive a shared secret key directly from their common but continuously fluctuating radio environment [7, 8]. The method's feasibility relies on the following observations [9]. First, all mobile wireless devices come equipped with radios that can sense their immediate radio environment. Second, co-located devices that simultaneously monitor a common set of ambient radio sources (*e.g.*, WiFi access points or cell phone base stations) will perceive very similar small-scale temporal variations in their wireless channels, known as *small-scale fading*. However, an adversary far from these co-located devices will perceive a very different small-scale fading. Third, due to the environmental factors, the radio channel varies in unpredictable ways over short time scales, which produces inherent randomness in secret key generation.

Based on the above observations, some practical key extraction protocols have been proposed. The received signal strength (RSS) is first used for key extraction because it can be easily perceived by wireless devices [10–13]. However, such methods cannot work well in static scenarios due to the infrequent and small-scale variations characterizing RSS measurements. In addition, such methods are vulnerable to predicable channel attack [10] because RSS readings will increase and decrease if the channel is blocked periodically. To address these issues, recent studies have proposed to extract secret keys using channel state information (CSI) available from orthogonal frequency-division multiplexing (OFDM) [14–16]. In contrast to RSS, CSI is a fine-grained metric derived from the physical layer. It consists of 56 subcarriers in the frequency domain and can be utilized to achieve higher generation bit rate [17].

In existing CSI-based key extraction protocols, the basic procedures are similar. The first step is *random bit generation*, where a quantization method is used to convert the CSI measurements to information bitstreams. Although the CSI measurements of two end-parties in close physical proximity are similar, the bitstreams generated respectively by them may not be consistent. Next, the inconsistent bits in these two bitstreams have to be found and removed to make these two bitstreams consistent, a process called *information reconciliation*. The error correcting technique is a common method used in information reconciliation. In this technique, partial information about the bitstreams is exchanged to facilitate error correcting. Finally, due to the information leakage in the process of information reconciliation [18], a procedure called *privacy amplification* [19] is implemented to protect the confidentiality and privacy of key generation.

Research motivation Existing CSI-based key extraction approaches can only work effectively when the distance between two co-located devices is very short (approximately 1 cm), which hampers their application in practice. This is because it is limited by the bounded error correcting capability of ECC used in information reconciliation procedures. According to our experiments, if we use an error-correcting-based approach, the pairing success rate will decease rapidly when the distance between the antennas of two devices exceeds 1 cm. We define the distance range that allows two devices to be successfully paired with a high probability as the *authentication distance*. Our research motivation comes from the question of how to increase the authentication distance without decreasing the authentication reliability.

Our contributions In this paper, we design a CSI-based authenticated key agreement protocol named PSP. PSP can achieve robust secure pairing of two wireless devices located in physical proximity. The intuition behind PSP is that the similarity degree of the CSI measurements of two co-located devices can be taken as the proof of physical proximity, and the consistent part of their CSI measurements can be used for shared key generation. Specifically, using PSP, two co-located devices that do not share a prior trust relationship first generate bitstreams according to their similar CSI measurements sampled from the same WiFi source. Then, rather than making these two bitstreams identical through information reconciliation, the bitstreams are convert into sets, and then, PSI computations are implemented to make the two devices obtain the intersection of these two sets. Finally, the cardinality of the intersection is used for authentication; then, the intersection itself is used to generated the symmetric key. In summary, PSP provides the following unique advantages.

– PSP enables a longer authentication distance. According to our experimental results, using an error-

correcting-based information reconciliation process, the authentication distance between two paring devices is approximately 1 cm. Under the same experimental conditions, the authentication distance of PSP is approximately 5 cm.
- Using an interactive error correcting method for information reconciliation [20], the secret bit generation rate will decrease rapidly with increasing distance. In contrast, PSP can maintain a stable secret bit generation rate within the authentication distance.
- In existing solutions, partial sensitive information is exchanged in plain text for information reconciliation. In PSP, all sensitive information is encrypted, which means reduced information leakage.

Organization The remainder of this paper is organized as follows. Section 2 introduces the related work in greater detail. Section 3 presents the necessary preliminaries and shows the feasibility of using CSI for secure pairing. Section 4 describes the system model and the design goals. Section 5 details the design of PSP. We provide the analysis of security and performance of PSP in Sect. 6. Section 7 shows the evaluation results on the implementation of PSP. Finally, we conclude our work in Sect. 8.

2 Related works

2.1 Secure pairing

In order to ensure communication security, it is imperative to form secure associations between wireless devices before formal data transmission. This process is generally called secure pairing. By implementing secure pairing, two devices can authenticate each other and exchange the communication keys. Compared to wired networks, achieving secure paring in wireless networks is more challenging. That is because the pairing in wireless networks is more vulnerable to eavesdropping attacks and man-in-the-middle attacks due to its open-air nature and the lack of security infrastructure. Facing this challenge, a lot of authentication and key exchange schemes for various kinds of wireless networks such as VANETs and GSM [21–24] systems have been proposed. These schemes are mainly based on the novel public key cryptosystems, such as identity-based encryption (IBE) or certificate less-public key cryptography (CL-PKC), and the authentication is based on user's ID or pseudonym. In this paper, we focus on the authentication and key agreement between two co-located and direct-connected mobile devices. Their similar ambient radio environment is taken advantage of to implement authentication and key generation.

2.2 Privacy protection

Privacy protection is an important issue that needs to be carefully considered when designing a secure protocol. Generally, privacy refers to the private information that should be exposed to other users or servers as little as possible. Such information can be divided into two categories. The first category is associated with user's sensitive personal information, and its leakage will allow the user to be identified and tracked by attackers. For example, the user's location is the most important private information in location-based services (LBSs). How to protect the location privacy has become the focus of attention in recent years [25–27]. The second category is associated with the secure pairing. The disclosure of such information will affect the reliability of the authentication and the security of the key. In this paper, the CSI measurement belongs to the second category of private information. The disclosure of the CSI measurement will cause the false pairing and key exposure. Therefore, we adopt PSI algorithm to prevent the disclosure of the devices' CSI measurements when designing PSP.

2.3 Key extraction using RF signal

Taking advantage of the randomness of a radio channel's physical layer characteristics for key extraction has been theoretically explored in a number of studies. The works done by Maurer [28] and Ahlswede and Csizar [29] show that correlated randomness can be used to generate secret keys, which establish the analytical foundations for secret key generation in wireless communication. Sayeed and Perrig [30] and Wilson et al. [31] exploit the randomness of phase for secret key extraction in OFDM and UWB systems, respectively. Wang et al. [32] propose a phase-based scalable and efficient secret key generation scheme. Tope et al. [33] utilize the randomness of the received signal's envelope to share the secrecy between two parties. However, all these investigations are based on theoretical analysis and only provide simulation results.

Because RSS can be easily measured using existing wireless infrastructures, it is used in most practical secret key generation methods that exploit temporal and spatial variations in the radio channel [10–13]. They tend to transform the RSS values into a sequence of bits and create secrets based on the reconciled bits. However, RSS can only provide coarse-grained channel information and may vary at different receivers; therefore, the key generation rate of RSS-based methods is low. Channel response has also been exploited to generate keys. For instance, Mathur et al. [34] utilize the channel impulse response (CIR) extracted from a single frequency to generate secret bits.

Compared to RSS, CSI is a much richer source of secret information and can be obtained via orthogonal frequency-division multiplexing (OFDM). The feasibility of using CSI in OFDM systems to generate secret keys has been explored in [16, 35], who establish the theoretic basis but do not provide any practical solution. Then, a practical CSI-based key extraction system is proposed by Liu et al. in [14].

Remarkably, the above-mentioned works mainly focus on key extraction. They are built on the assumption that two communicating devices have been authenticated previously, whereas our scheme provides both authentication and security. The work that is best related to our work is proposed by Mathur et al. in [7] and named ProxiMate. ProxiMate also provides authentication and security. The differences between PSP and ProxiMate are as follows. First, in ProxiMate, Alice and Bob generate the bitstreams by measuring common RF signals such as FM signals or TV signals. However, this is not currently possible using commercial off-the-shelf wireless devices, such as cell phones or WiFi cards, to measure these signals. In our scheme, we utilize the CSI as the proof of authentication and the source of the secret key. It is supported by 802.11n and 802.11ac and can be extracted using commodity off-the-shelf wireless NIC models. In addition, CSI contains richer information about the wireless channel than do FM or TV signals. Therefore, using CSI can significantly increase the secret bit generation rate. Second, ProxiMate uses (23,12) Golay code for information reconciliation. As we present in Sect. 6, it suffers from low fault tolerance and allows only a very short authentication distance. Compared to ProxiMate, PSP is based on PSI and allows a longer authentication distance.

3 Preliminaries

In this section, we present the background knowledge needed for understanding our work. First, basic information about CSI is introduced. Second, we illustrate the feasibility of using CSI data for key agreement. Finally, we show the basic principles of homomorphic encryption and PSI.

3.1 CSI model

802.11a/g/n adopt orthogonal frequency-division multiplexing (OFDM) technology. In OFDM, the overall wide bandwidth channel is divided into many small but orthogonal subcarriers. Thus, for OFDM systems, channel estimation is equivalent to measuring the parameters of all the subcarriers. Compared with one-dimensional temporal RSS data, the multi-dimensional CSI data contain more abundant channel characteristic information. In 802.11n and its successor, 802.11ac, CSI in each frame is a large complex-number matrix that describes the channel frequency response (CFR) for each subcarrier in every spatial stream. Its size is $N_{tx} \times N_{rx} \times k$, where N_{tx} and N_{rx} denote the number of transmitting and receiving antennas, respectively, and k is the number of subcarriers. Each complex value h in the CSI matrix can be transformed into polar coordinates as $h = |h|e^{j\angle h}$, where h and $\angle h$ denote the amplitude and phase of each subcarrier. Due to its spatial decorrelation property, both the amplitude and phase of the CSI value can be used for key extraction. In this paper, we will focus on the amplitude of the CSI value.

3.2 The feasibility of using CSI for key agreement

In this paper, we use CSI as the proof of physical proximity and the source of secret information. Its feasibility is supported by the following properties. First, CSI is a unique and correlated measurement for devices at a particular physical location. Second, CSI measurements sampled by different devices are rapidly decorrelated with the distance between the devices. Third, CSI is unpredictable due to its excellent random property caused by the multi-path effect of signal propagation.

According to the above-mentioned properties, if two devices are located within a short distance, their CSI measurement on the same WiFi source will be very similar to each other. As a general rule of thumb, such distance should be less than half of the wavelength of WiFi transmission [7]. For example, using a 2.4 GHz WiFi signal (wavelength $\lambda = 12.5$ cm), the distance between two co-located devices should be less than 6.25 cm. In this paper, we call 0.5λ the *legitimate distance*. If the distance between two co-located devices exceeds 0.5λ, their CSI measurements will be uncorrelated.

We conduct experiments to validate our proposal. In our experiments, we use four laptops [named Alice, Bob, Eve and Peter (AP)] with the Linux 802.11n CSI Tool [36] and Intel 5300 wireless NICs installed to form an AP-Client network operating in the 802.11n 2.4 GHz channel. We set the distance between the antennas of Alice and Bob to be 0.1λ, and we set the distance from Eve to both Alice and Bob to be 1.5λ. Here, the wavelength $\lambda = 12.5$ for the 2.4 GHz WiFi signal. Alice, Bob and Eve are first synchronized; then, they collect measurements of CSI amplitude values from the data frames broadcasted by Peter independently.

For each device (Alice, Bob and Eve), we collect 600 data frames broadcasted by Peter. Then, we obtain the amplitude values of all 30 subcarriers of each data frame from the CSI matrix contained in the frame header. The

experiment results are displayed in Fig. 1(a–c). Taking Fig. 1(a) for example, we find the subcarrier CSI amplitude variation sampled from the 600 data frames. Specifically, the z-axis (CSI Gain) represents the CSI amplitude value, the x-axis represents the subcarrier index, and the y-axis (Packet Index) represents the data frame index. To be more intuitive, different colors are used to render the amplitude variation. As we can see, because Alice and Bob are located in close physical proximity (Fig. 1a, b), their measured CSI values look very similar. However, Eve, which is far from Alice and Bob, shows much different measured CSI values (Fig. 1c).

In addition, based on the hardware used in our experiments, we investigate the correlation of CSI measurements among different subcarriers. Figure 1(d) plots the correlation of CSI measurements among 30 subcarriers in static scenarios. The figure shows that the CSI measurements have strong correlation between two adjacent subcarriers. For adjacent subcarriers, the correlations of CSI values are usually greater than 0.8. Although the subcarrier signals in an OFDM channel are orthogonal with different frequencies, the adjacent subcarriers have very similar frequencies, which results in similar channel responses at the receiver in the frequency domain. Based on this observation, if we use CSI measurements of all the subcarriers, the generated key may have many identical segments, which could be a risk factor under key cracking. Therefore, in this work, we only select a portion of the subcarriers that have low correlation between each other for key agreement.

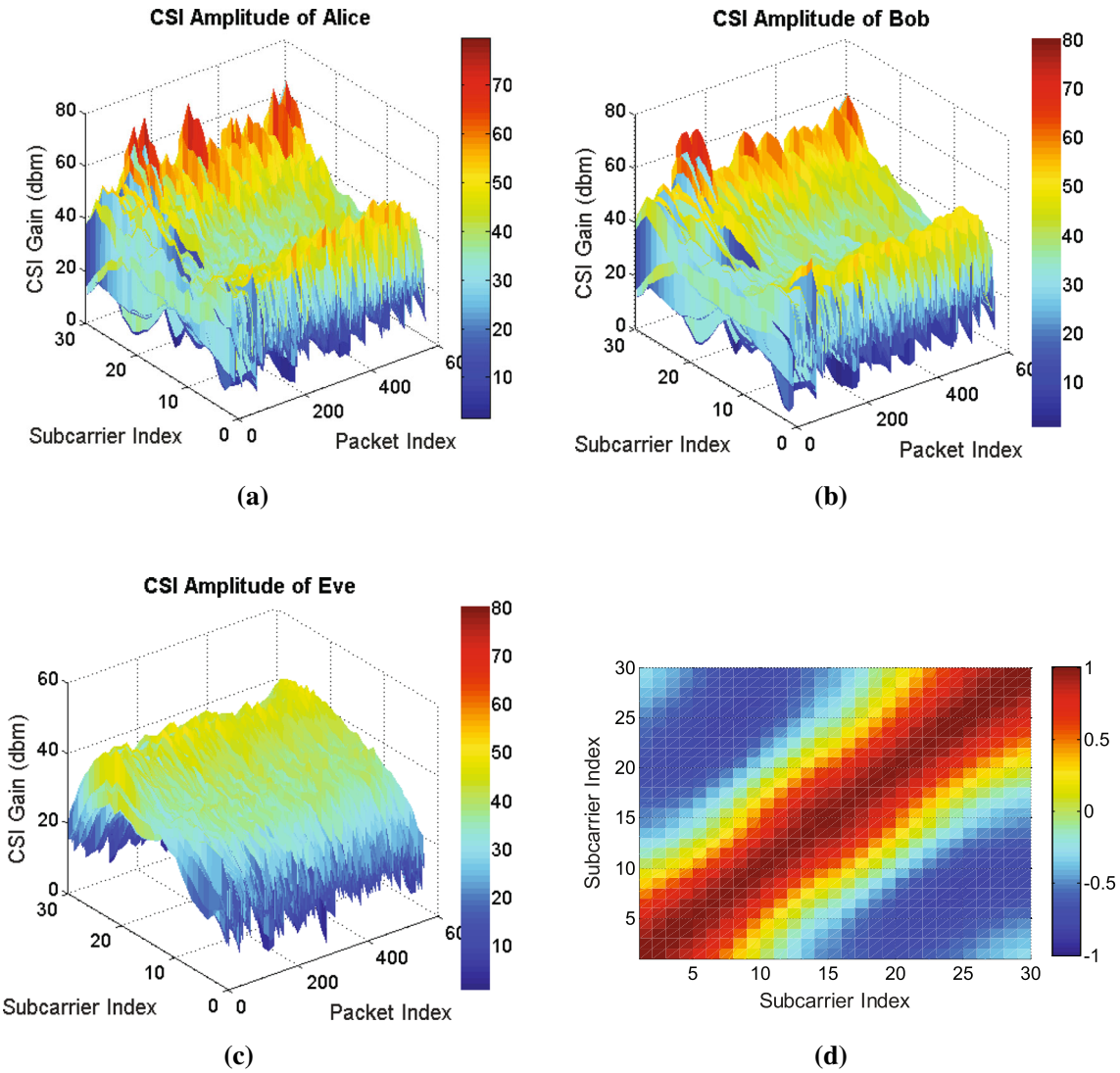

Fig. 1 The feasibility of using CSI. **a** CSI amplitude of Alice, **b** CSI amplitude of Bob, **c** CSI amplitude of Eve, **d** CSI measurements correlation among different subcarriers

3.3 Homomorphic encryption and PSI

Homomorphic encryption is an effective solution to privacy-preserving computation. It allows certain algebraic operations on the plaintext to be performed using (possibly different) algebraic operations directly on the ciphertext. There are several homomorphic cryptosystems [37–39]. We use *Paillier's cryptosystem* [39] in our work, which is simple, efficient and widely used. This cryptosystem supports the following operations, which can be performed without knowledge of the private key:

$$E(m_1, r_1) \cdot E(m_2, r_2) = E(m_1 + m_2, r_1 r_2) \ mod \ n^2 \quad (1)$$

$$E(m_1, r_1)^{m_2} = E(m_1 \cdot m_2, r_1^{m_2}) \ mod \ n^2 \quad (2)$$

Note that the random number r in a ciphertext $E(m, r)$ does not contribute to decryption or other homomorphic operations. It only prevents dictionary attacks by randomizing the ciphertext. For the sake of simplicity, we use $E(m)$ instead of $E(m, r)$ in the remainder of the paper. The above properties can be simplified as $E(m_1) \cdot E(m_2) = E(m_1 + m_2)$, $E(m_1)^{m_2} = E(m_1 \cdot m_2)$.

Private set intersection (PSI) is a useful cryptographic primitive that allows two parities (*client* and *server*) to interact based on their respective (private) input sets in such a way that the *client* obtains nothing other than the set intersection, and the *server* learns nothing beyond the client set size. Based on Paillier's cryptosystem, Freedman, in [40], proposed an efficient private set intersection protocol, which is applied in our work. In this protocol, the *client* defines a polynomial P whose roots are his input set elements:

$$P(y) = (x_1 - y)(x_2 - y)\ldots(x_k - y) = \sum_{u=0}^{k} a_u y^u$$

Then, he sends the homomorphic encryptions of the coefficients of this polynomial to the *server*. The *server* uses the homomorphic properties of the encryption system to evaluate the polynomial at each element of his private set. He then multiplies each result by a fresh random number r to obtain an intermediate result, and he adds to it an encryption of the value of his input, i.e., the *server* computes $E(r \cdot P(y) + y)$. Therefore, for each of the elements in the intersection of the two party input sets, the result of this computation is the value of the corresponding element, whereas for all other values, the result is random. The details can be found in Sect. 5.

4 System model and design goals

4.1 System model

As we can see from Fig. 2, our system model consists of four wireless devices: Peter, Alice, Bob and Eve. Peter is a WiFi AP. Alice and Bob are two legitimate devices that are located within the *legitimate distance* (0.5λ, 6.25 cm for WiFi at 2.4 GHz). Eve is a malicious device and is located beyond the legitimate distance (far more than 0.5λ) to both Alice and Bob. Peter, Alice, Bob and Eve form an AP-Client network operating in the 802.11n 2.4 GHz channel.

In our system model, Alice and Bob have no prior shared secret. They want to authenticate each other by taking the CSI measurements sampled from the frames transmitted by Peter as the proof of physical proximity. When the authentications are passed in both sides, their CSI measurements are further used to generate a shared secret key for secure communication between them.

4.2 Threat model

In this paper, we discuss the security issue based on the honest-but-curious (HBC) model. In the HBC model, each party (Alice, Bob and Eve) could attempt to learn more information than allowed by inferring from the results but honestly following the protocol. In addition, we give the following assumptions. First, all the details of our key agreement protocol (including the values of all the parameters) are public. Second, Eve can eavesdrop on all

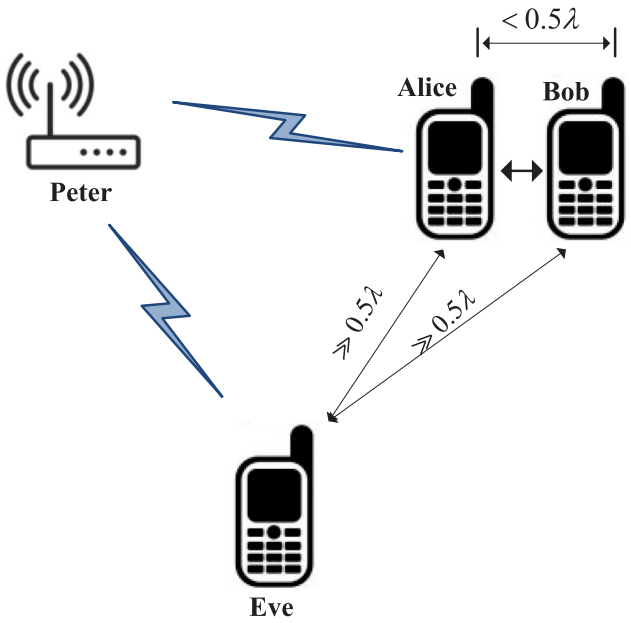

Fig. 2 System model

communications between Alice and Bob. In this paper, we mainly consider the following two malicious attacks, which should be well resisted in our scheme.

1. *Masquerade attack* Eve pretends to be the legitimate device and attempts to establish secure association with Alice or Bob.
2. *Key cracking attack* Eve eavesdrops on the communications between Alice and Bob and attempts to crack their shared key by taking advantage of the collected information.

4.3 Design goals

1. *Realizing bidirectional authentication* The authentication between Alice and Bob should be bidirectional. Only located within the legitimate distance, the authentication between two devices is likely to succeed.
2. *Resisting masquerade attacks* The confidentiality of the private information of Alice and Bob, such as the CSI measurements and the generated bitstreams, should be guaranteed. There should be minimal information leakage when Alice and Bob exchange authentication information. Eve cannot obtain effective information to implement a masquerade attack.
3. *Resisting key cracking attacks* The shared secret key with arbitrary length generated by our protocol should have strong randomness to resist brute force attacks.
4. *Allowing longer authentication distances* Even within the legitimate distance, the consistency of the CSI measurements of two co-located devices will decrease rapidly with increasing distance between them, which may cause an authentication failure. We define the distance range that allows two devices to be successfully authenticated with a high probability as the *authentication distance*. Considering the usability, our protocol should make the authentication distance as close as possible to the legitimate distance (0.5λ).

5 Our design

5.1 Overview

Figure 3 shows the design model of PSP, which consists of three main procedures: *bitstream generation*, *encoding*, *authentication and key generation*.

1. *Bitstream generation* Alice and Bob sample signals from the same WiFi source and obtain a sequence of CSI measurements S_a and S_b. Then, by quantifying S_a and S_b, Alice and Bob obtain 0–1 bitstreams B_a and B_b, respectively.
2. *Encoding* By combining two adjacent bits and adding the index information, B_a and B_b are converted to the data set $\hat{B}_a$ and $\hat{B}_b$, which can be used for the subsequent PSI-based authentication and key generation.
3. *Authentication and key generation* Using $\hat{B}_a$ and $\hat{B}_b$, Alice and Bob authenticate each other based on PSI and further generate the shared key.

Below, we expand and elaborate upon each of the steps described above. Refer to Table 1 for a summary of the major notation used.

5.2 Bitstream generation and encoding

5.2.1 Subcarrier selection

According to the observations shown in Fig. 1(d), the CSI measurements of two adjacent subcarriers are strong correlated. Therefore, it is unnecessary to convert the CSI measurements of all subcarriers to bitstreams. In this work,

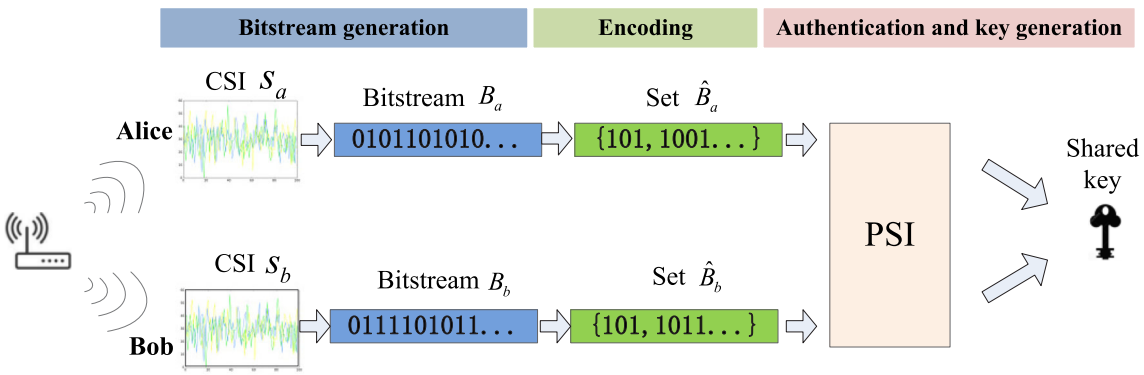

Fig. 3 The main process of PSP

Table 1 Summary of the major notation

Symbol	Meaning
S_a, S_b	The CSI measurement sequences sampled by users Alice (S_a) and Bob (S_b)
B_a, B_b	The bitstreams generated by Alice (B_a) and Bob (B_b) quantifying S_a and S_b
$\hat{B}_a, \hat{B}_b$	The sets generated by encoding B_a and B_b
PU_a, PR_a	The key pair generated by Alice. PU_a is public, PR_a is private
$P_a(y)$	The polynomial generated by Alice according to $\hat{B}_a$
τ	The predefined threshold used for authentication
K_{ab}	The consistent part of B_a and B_b

we only select portions of the subcarriers of which the CSI measurements are weakly correlated. Specifically, we use the CSI measurements of the 1st, 10th and 24th subcarriers for bitstream generation.

5.2.2 *Quantization*

The sampled CSI measurements have to be quantified to be converted into 0–1 bitstreams. In our work, we use the quantizer mentioned in [10], which is efficient and accurate.

Suppose that, at a time sequence $[t_1, t_2, \ldots, t_n]$, the channel measurements sampled by Alice and Bob are two amplitude sequences of CSI: $S_a(t_1, \ldots, t_n)$ and $S_b(t_1, \ldots t_n)$. The quantizer can be described as follows:

1. Each user (Alice and Bob) calculates two adaptive thresholds q_+ and q_- independently

$$\begin{cases} q_+ = \mu_{S(t_1,\ldots,t_n)} + \alpha * \sigma_{S(t_1,\ldots,t_n)} \\ q_- = \mu_{S(t_1,\ldots,t_n)} - \alpha * \sigma_{S(t_1,\ldots,t_n)}, \end{cases}$$

   where μ and σ are the mean and standard deviation of $S(t_1, \ldots, t_2)$, respectively, and $\alpha \geq 0$ is a tuning constant.

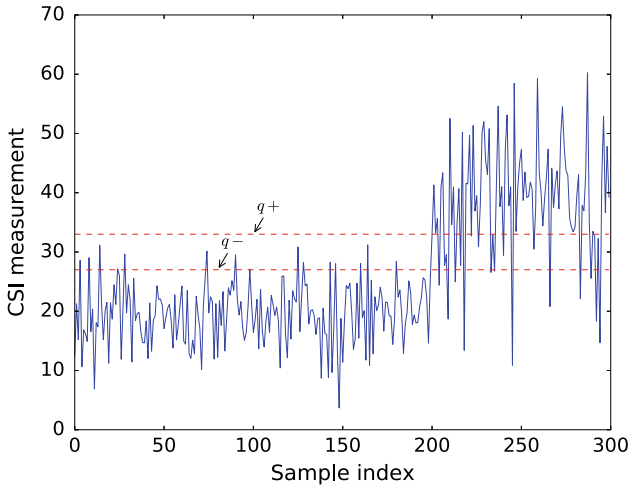

Fig. 4 The impact of q

2. Alice and Bob parse S_a and S_b, respectively; discard those CSI estimates that lie between q_+ and q_-; and maintain a list of indices to track the CSI estimates that are discarded. They exchange their lists of dropped CSI estimates and only keep the estimates that they both decide not to drop.
3. Alice and Bob generate their bitstreams by extracting a "1" or a "0" for each CSI estimate if the estimate lies above q_+ or below q_-.

For the quantizer mentioned above, the challenge is to choose the proper thresholds q_+ and q_-. If fixed values of q_+ and q_- are used to quantify the whole sequence of CSI measurements, this may lead to an undesired bitstream that has low entropy and that is easy to crack. For instance, as we can see in Fig. 4, the two red dashed lines denote the thresholds q_+ and q_- calculated from all the CSI measurements ($\alpha = 0.3$). Most values with a sample index $<$ 200 are smaller than q_-, whereas the values with a sample index $>$ 200 are mostly larger than q_+. If these two fixed threshold values are used for quantization, there will be a long run of zeros in the front part of the bitstream and a long run of ones in the rear part. Therefore, to obtain bitstreams with strong randomness, we let each user divide their CSI measurements into small blocks and calculate q_- and q_+ for each block.

The next question is how large the block size should be. We conduct experiments to determine the appropriate block size. In our experiments, we set the distance between Alice and Bob to be 1 cm, and we let them simultaneously gather 232 samples of CSI measurements. The sampling interval is set to be 200 ms, and $\alpha = 0.3$. Then, we compare the generated bitstreams. Figure 5 shows the comparison results. As we can see, a too small of a block size will increase the number of inconsistent bits. This is because some outlier samples with extremely large or small values have a serious impact on the mean value of CSI in one block, which leads to an inappropriate threshold q_+ and q_-. However, too large of a block size will reduce the randomness of the generated bitstreams. To obtain a good trade-off, in our work, we set the block size as $k = 40$.

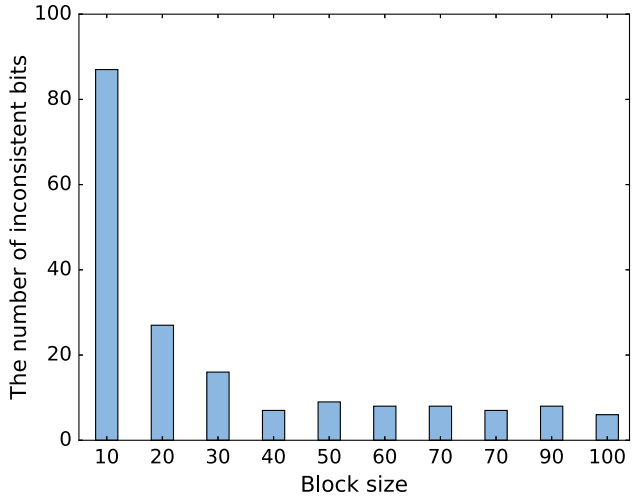

Fig. 5 The impact of block size k

5.2.3 Combination and encoding

After converting the CSI measurements of selected subcarriers to 0–1 bitstreams, Alice and Bob concatenate these bitstreams together to obtain B_a and B_b, respectively. In our work, $B_a = B_{a,1}||B_{a,10}||B_{a,24}$ $B_b = B_{b,1}||B_{b,10}||B_{b,24}$. Here, $B_{a,i}$ ($B_{b,i}$) means the bitstream generated from the ith subcarrier, and the symbol || means concatenating two sequences.

However, the bitstreams B_a and B_b cannot be directly used for authentication and key generation. This is because B_a and B_b only contain two types of values (0 and 1). Therefore, they cannot be treated as sets to be used for PSI computation. The simplest way to address this problem is to add an index number for each bit in B_a and B_b. For example, let $B = [1, 0, 1, 1, 1, 0, 1, 0]$ be the original bitstream. Now, we add the index number for each bit in B to convert B to $\hat{B} = [\underline{1}1, \underline{1}00, \underline{1}11, \underline{1}001, \underline{1}011, \underline{1}100, \underline{1}111, \underline{1}0000]$. For each element in $\hat{B}$, the part with an underline stands for its index. Obviously, because each component now is unique, $\hat{B}$ can be treated as a set to be used for PSI computation.

However, only adding an index for each bit in the bitstream is not secure. That is because Alice (as the initiator of the authentication) can infer the whole bitstream of Bob from $\hat{B}_a \cap \hat{B}_b$. For example, let $\hat{B}_a = [\underline{1}1, \underline{1}00, \underline{1}10, \underline{1}001, \underline{1}010, \underline{1}100]$, $\hat{B}_b = [\underline{1}0, \underline{1}01, \underline{1}10, \underline{1}000, \underline{1}011, \underline{1}101]$. If Alice knows $\hat{B}_a \cap \hat{B}_b = \underline{1}10$, she can infer that the bit value $B_a[i]$ in other positions except position 3 is opposite to $B_b[i]$. In other words, $B_a[i] = \overline{B_b[i]}$ when $i \neq 3$. Therefore, although there is only one component in the intersection, Alice can learn the whole B_b through it.

Let $\hat{B}_a$ and $\hat{B}_b$ be the sets generated from B_a and B_b. According to the above analysis, B_a and B_b should not be inferred from $\hat{B}_a \cap \hat{B}_b$. To this end, we generate $\hat{B}_a$ and $\hat{B}_b$ through two successive steps. In the first step, we convert B_a (B_b) to $B_a^{'}$ ($B_b^{'}$). $|B_a^{'}| = \frac{1}{2}|B_a|$ ($|B_b^{'}| = \frac{1}{2}|B_b|$) and $B_a^{'}[i] = B_a[2i-1]||B_a[2i]$ ($B_b^{'}[i] = B_b[2i-1]||B_b[2i]$). For example, if $B_a = [0, 1, 1, 0, 1, 0]$, then $B_a^{'} = [01, 10, 10]$. In the second step, we add the index number for each component in $B_a^{'}$ ($B_b^{'}$), and then, we convert it to $\hat{B}_a$ ($\hat{B}_b$). $\hat{B}_a[i] = i||B_a^{'}[i]$ ($\hat{B}_b[i] = i||B_b^{'}[i]$). For example, if $B_a^{'} = [01, 10, 10]$, then $\hat{B}_a = [\underline{1}01, \underline{1}010, \underline{1}110]$. The whole process of bitstream generation and encoding is shown in Fig. 6.

5.3 Authentication and key generation

In this procedure, $\hat{B}_a$ and $\hat{B}_b$ are further taken as the inputs of the PSI computation for authentication and key generation. Specifically, *authentication* here means that both Alice and Bob attempt to confirm that she/he is really interacting with the one she/he intends to communicate with (the one located in physical proximity). In our work, this is achieved by computing $\hat{B}_a \cap \hat{B}_b$ using the PSI-based approach. According to our system model and the preliminary observations (Fig. 2), the CSI measurements sampled by Alice and Bob should be similar. Our basic idea is to measure the similarity degree of Alice's and Bob's CSI measurements by the value of $|\hat{B}_a \cap \hat{B}_b|$. Intuitively, the more similar S_a and S_b are, the larger the value of $|\hat{B}_a \cap \hat{B}_b|$ is. Therefore, Alice and Bob can authenticate each other if both of them learn that the value of $|\hat{B}_a \cap \hat{B}_b|$ exceeds an appropriately determined threshold. When the authentication is successful, $\hat{B}_a \cap \hat{B}_b$ can be further used to generate the shared key. The details are presented as follows.

Protocol 1: The basic authentication and key generation protocol

Input: $\hat{B}_a = [x_1, x_2, \ldots x_n]$, $\hat{B}_b = [y_1, y_2, \ldots, y_n]$

Step 1: Alice performs the following in sequence.

a. Alice generates the Paillier key pair PU_a and PR_a. PU_a is the public key, which is used for encryption, and PR_a is the private key, which is used for decryption.
b. Alice defines a polynomial P_a whose roots are the elements in $\hat{B}_a$ and uses interpolation to compute the coefficients of P_a.

$$\begin{aligned} P_a(y) &= (x_1 - y)(x_2 - y)\ldots(x_n - y) \\ &= \sum_{k=0}^{n} a_k y^k \end{aligned}$$

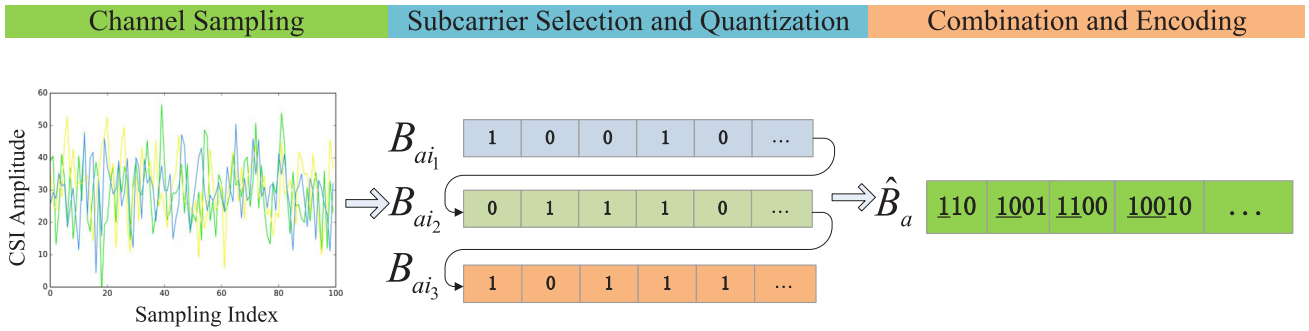

Fig. 6 The process of bitstream generation

c. Using PU_a, Alice encrypts each of the $(n+1)$ coefficients by the semantically secure homomorphic encryption scheme and sends to Bob PU_a and the ciphertext vector $\mathbf{P}_a = [E_{PU_a}(a_0), E_{PU_a}(a_1), \ldots E_{PU_a}(a_n)]$.

Step 2: Bob performs the following after receiving PU_a and $\mathbf{P}_a$.

a. Bob uses the homomorphic properties to evaluate the value of the polynomial $P_a(y)$ at y_i, that is, for each y_i in $\hat{B}_b$, he computes $E_{PU_a}(P_a(y_i))$. Specifically,

$$\begin{aligned} E_{PU_a}(P_a(y_i)) &= E_{PU_a}\Big(\sum_{k=0}^{n} a_k y_i^k\Big) \\ &= E_{PU_a}(a_0 + a_1 y_i^1 + a_2 y_i^2 + \cdots + a_n y_n^n) \\ &= E_{PU_a}(a_0) \cdot E_{PU_a}(a_1 y_i^1) \ldots \cdot E_{PU_a}(a_n y_i^n) \\ &\quad [according\ to\ Eq.(1)] \\ &= E_{PU_a}(a_0) \cdot E_{PU_a}(a_1)^{y_i^1} \ldots \cdot E_{PU_a}(a_n)^{y_i^n} \\ &\quad [according\ to\ Eq.(2)] \\ &= \prod_{k=0}^{n} E_{PU_a}(a_i)^{y_i^k}. \end{aligned}$$

Then, Bob generates a random value r and computes $E_{PU_a}(rP_a(y_i) + y_i)$ for each $E_{PU_a}(P_a(y_i))$. Specifically,

$$\begin{aligned} &E_{PU_a}(rP_a(y_i) + y_i) \\ &\quad = E_{PU_a}(P_a(y_i))^r \cdot E_{PU_a}(y_i) \end{aligned}$$

c. Bob randomly permutes the set $R_{ab} = \{E_{PU_a}(rP_a(y_1) + y_1), E_{PU_a}(rP_a(y_2) + y_2), \ldots, E_{PU_a}(rP_a(y_n) + y_n)\}$. Then, he sends the permuted R_{ab} to Alice.

Step 3: Alice performs the following after receiving R_{ab}.

a. Alice uses PR_a to decrypt each element in R_{ab}. Specifically, Alice computes $D_{PR_a}(E_{PU_a}(rP_a(y_i) + y_i))$. Then, Alice obtains the plaintext set $\hat{R}_{ab} = \{rP_a(y_1) + y_1, rP_a(y_2) + y_2, \ldots, rP_a(y_n) + y_n\}$.

b. Alice computes $I_{ab} = \hat{B}_a \cap \hat{R}_{ab}$. If $|I_{ab}|$ exceeds the predefined threshold τ, she is convinced that Bob is the device located within the legitimate distance range. If $|I_{ab}|$ is less than τ, the authentication fails, and the protocol is aborted.

c. If the authentication succeeds, Alice extracts the index information of all the elements in I_{ab} to generate the set $T = \{Ind(I_{ab}[i]) | i = 0, 1, \ldots, |I_{ab}| - 1\}$. Here, $Ind(I_{ab}[i])$ refers to the index number of the i-th element of I_{ab}. For example, if $I_{ab} = \{\underline{101}, \underline{1010}, \underline{1101}, \underline{10010}\}$, then $T = \{10, 101, 110, 1001\}$.

d. Alice extracts the data part of all the elements in I_{ab} and splices them together in order to generate a new bitstream K_{ab}, which can be used as the shared key. For example, if $I_{ab} = \{\underline{101}, \underline{1110}, \underline{11011}, \underline{100001}\}$, then $K_{ab} = 01101101$.

e. Alice generates a random number N. Then, using a predefined symmetric encryption algorithm (such as DES or AES) and taking K_{ab} as the key, Alice encrypts N. Finally, Alice sends N, T, and $E_{K_{ab}}(N)$ to Bob.

Step 4: Bob performs the following after receiving Alice's message.

a. If $|T| > \tau$, Bob chooses corresponding elements in $\hat{B}_b$ according to T, extracts their data parts and splices them together, as Alice does to obtain a new bitstream K'_{ab}. If $T < \tau$, the authentication fails, and the protocol is aborted.

b. Bob attempts to decrypt $E_{K_{ab}}(N)$ by K'_{ab}. If the result is equal to N(i.e.$K'_{ab} = K_{ab}$), Bob successfully authenticates Alice, which means that Bob is convinced that Alice is the legitimate device. Finally, K_{ab} can be used for further encrypted communication between Bob and Alice.

5.4 Reducing the polynomial degree

The above-mentioned protocol shows the basic steps of the authentication and key generation procedure. However, it cannot be directly used in practice. That is because the degree of the polynomial defined by Alice according to $\hat{B}_a$ is excessive. In the actual computations, the high degree of the polynomial will lead to an overflow and result in an incorrect output. To reduce the polynomial degree, we make Alice and Bob divide $\hat{B}_a$ and $\hat{B}_b$ into small groups and define a polynomial for each group. The modified protocol is described as follows.

Protocol 2: The modified authentication and key generation protocol
Input: $\hat{B}_a = [x_1, x_2, \ldots x_n]$, $\hat{B}_b = [y_1, y_2 \ldots, y_n]$
Step 1: Alice performs the following in sequence.

a. Alice generates the Paillier key pair PU_a and PR_a.
b. Alice divides the elements in $\hat{B}_a$ into m groups, where each group contains υ elements. Here, we assume that n is divisible by m, and $n/m = \upsilon$. Now, Alice obtains $\hat{B}_a^{'}$,

$$\hat{B}_a^{'} = [X_1, X_2, X_3, \ldots, X_m],$$
$$X_i = [x_{(i-1)\upsilon+1}, x_{(i-1)\upsilon+2}, \ldots, x_{i\upsilon}],$$

c. For each group X_i, Alice defines a polynomial P_{X_i}, whose roots are the elements in X_i.

$$P_{X_i}(y) = (x_{(i-1)\upsilon+1} - y)$$
$$(x_{(i-1)\upsilon+2} - y)\ldots(x_{i\upsilon} - y) = \sum_{k=0}^{\upsilon} a_{i,k} y^k$$

d. Let $\mathbf{a}_i = [a_{i,0}, a_{i,1}, \ldots, a_{i,\upsilon}]$ be the coefficient vector of P_{X_i}. Using PU_a, Alice encrypts the $(\upsilon + 1)$ elements in each $\mathbf{a}_i$ and then obtains $E_{PU_a}(P_{X_i}(y)) = [E_{PU_a}(a_{i,0}), E_{PU_a}(a_{i,1}), \ldots, E_{PU_a}(a_{i,\upsilon})]$
e. Let $\mathbf{P}_a = [E_{PU_a}(P_{X_1}(y)), E_{PU_a}(P_{X_2}(y)), \ldots, E_{PU_a}(P_{X_m}(y))]$. $\mathbf{P}_a$ is sent to Bob by Alice.

Step 2: Bob performs the following after receiving PU_a and $\mathbf{P}_a$.

a. Similarly to what Alice does, Bob divides the elements in $\hat{B}_b$ into m groups, where each group contains υ elements, and then obtains $\hat{B}_b^{'}$,

$$\hat{B}_b^{'} = [Y_1, Y_2, Y_3, \ldots, Y_m],$$
$$Y_i = [y_{(i-1)\upsilon+1}, y_{(i-1)\upsilon+2}, \ldots, y_{i\upsilon}].$$

b. Bob generates a random value r. For each $E_{PU_a}(P_{X_i}(y)) \in \mathbf{P}_a$ and each $y_k \in Y_i$, Bob computes $E_{PU_a}(rP_{X_i}(y_k) + y_k)$, and then, he generates the set R_{ab} (see the details in Algorithm 1).
c. Bob randomly permutes R_{ab} and then sends it to Alice.

Step 3: Alice performs the following after receiving R_{ab}.

a. Alice uses PR_a to decrypt all n ciphertexts received in R_{ab} and obtains the plaintext set $\hat{R}_{ab} = \{rP_{X_i}(y_k) + y_k) | i = 1, 2, \ldots m, k = (i-1)\upsilon + 1, (i-1)\upsilon + 2, \ldots, i\upsilon\}$.
b. Alice computes $I_{ab} = \hat{B}_a \cap \hat{R}_{ab}$. If $|I_{ab}|$ is less than τ, the authentication fails, and the protocol is aborted.
c. If the authentication succeeds, Alice extracts the index information of all the elements in I_{ab} to generate the set $T = \{Ind(I_{ab}[i]) | i = 0, 1, ..., |I_{ab}| - 1\}$.
d. Alice extracts the data part of all the elements in I_{ab} and splices them together according to their index number to generate a new bitstream K_{ab} as the shared key.
e. Alice generates a random number N and encrypts N according to a predefined symmetric encryption algorithm using K_{ab} as the key. Then, Alice sends N, T, and $E_{K_{ab}}(N)$ to Bob.

Step 4: Same as step 4 in Protocol 1.

Algorithm 1

Input:

$\hat{B}_b^{'} = [Y_1, Y_2, ..., Y_m]$
random value r
PU_a and $\mathbf{P}_a$

Output:

$R_{ab} = \{E_{PU_a}(rP_{X_i}(y_k) + y_k) | i = 1, 2, ..m, k = (i-1)\upsilon + 1, (i-1)\upsilon + 2, ..., i\upsilon\}$

```
R_ab = {}
for each Y_i ∈ B̂_b' do
    for each y_k ∈ Y_i do
        compute E_PUa(rP_Xi(y_k) + y_k)
        R_ab = R_ab ∪ {E_PUa(rP_Xi(y_k) + y_k)}
    end for
    return R_ab
end for
```

6 Analysis

6.1 Security analysis

In this section, we analyze the security of PSP based on the HBC model. As mentioned in Sect. 4.2, in the HBC model, PSP should be able to resist masquerade attack and key cracking attack. In order to achieve these two goals, we mainly focus on two security issues in our design. The first is authentication security, which contains two points. (i) The authentication should be bidirectional. (ii) Only the devices located within the legitimate distance can pass the authentication and generate the shared key. The second is key security. That is to say, the randomness and length of the shared key K_s can resist the brute force attack and dictionary attack from Eve. The following analysis shows that these two security goals can be effectively achieved in PSP.

6.1.1 Semantic security of the Paillier cryptosystem

We build our security analysis on the assumption that the *Paillier cryptosystem* is semantically secure [39].

Theorem 1 *The Paillier Cryptosystem is semantically secure if and only if the Decisional Composite Residuosity Assumption holds.*

Proof In [39], Paillier first defines the decisional composite residuosity problem. Then, he presents the conjecture that the decisional composite residuosity problem is intractable. Finally, he presents Theorem 1 and proves it by reducing the semantic security issue of the Paillier cryptosystem to the decisional composite residuosity problem. Due to space limitation, we direct the reader to [39] for a complete proof. □

6.1.2 Authentication security

For ease of analysis, we first give the diagram of the message sequence exchanged between Alice and Bob in Protocol 1. Then, based on Fig. 7 and the system model (Fig. 2) presented in Sect. 4.1, we present the properties of PSP in terms of the authentication security through a series of *Claim* and *Proof*.

Claim 1 *The authentication in PSP is bidirectional.*

Proof Obviously, Alice (the initiator of the PSI) can first authenticate Bob according to the $\hat{B}_a \cap \hat{B}_b$ that she learned through the PSI computation in PSP (by message 1 and 2 in Fig. 7). We say that the authentication implemented by Alice is *explicit*.

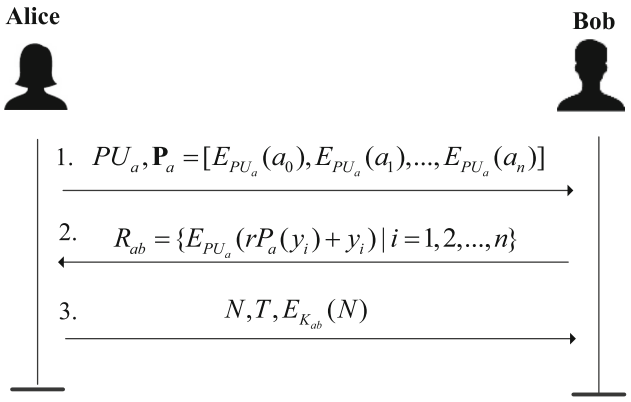

Fig. 7 Message sequence

As the responder of the PSI, Bob cannot directly learn the value of $\hat{B}_a \cap \hat{B}_b$. According to PSP, Bob has to use N, T, and $E_{K_{ab}}(N)$ (message 3 in Fig. 7) to authenticate Alice. Because T contains all the index numbers of the elements in $\hat{B}_a \cap \hat{B}_b$ ($|T| = |\hat{B}_a \cap \hat{B}_b|$), Bob can first compare $|T|$ with the threshold τ. If $|T| > \tau$, then Bob can further verify the authenticity of T. Specifically, Bob can derive K'_{ab} from the elements of $\hat{B}_b$ whose index number is contained in T. Then, he attempts to decrypt $E_{K_{ab}}(N)$. If $D_{K'_{ab}}(E_{K_{ab}}(N)) = N$, this implies that $K'_{ab} = K_{ab}$. Further, if $K'_{ab} = K_{ab}$, Bob can confirm the authenticity of T. Because T is authentic and $|T| > \tau$, Bob is convinced that Alice is the legitimate device. We say that the authentication implemented by Bob is *implicit*. According to the above analysis, the authentication in PSP is bidirectional. □

Claim 2 *If honestly following the PSP protocol, Alice cannot gain any additional information about $\hat{B}_b$ other than $\hat{B}_a \cap \hat{B}_b$.*

Proof As we can see in Fig. 7, the only message received by Alice from Bob is R_{ab} ($R_{ab} = \{E_{PU_a}(rP_a(y_i) + y_i)\}$). If $y_i \in \hat{B}_a \cap \hat{B}_b$, $P_a(y_i) = 0$. therefore, the decryption result is y_i. For $y_i \notin \hat{B}_a \cap \hat{B}_b$, the decryption result is a meaningless random number. Therefore, Alice cannot gain any additional information about $\hat{B}_b$ other than $\hat{B}_a \cap \hat{B}_b$. □

Claim 3 *If honestly following the PSP protocol, Bob cannot gain any additional information about $\hat{B}_a$ other than $\hat{B}_a \cap \hat{B}_b$.*

Proof As we can see in Fig. 7, Bob totally receives two messages from Alice. The first message contains PU_a and $\mathbf{P}_a = [E_{PU_a}(a_0), E_{PU_a}(a_1), \ldots E_{PU_a}(a_n)]$. According to Theorem 1, Bob cannot recover a_i from each $E_{PU_a}(a_i)$. Therefore, Bob cannot deduce $P_a(y)$ from this message,

let alone get $\hat{B}_a$. The second message contains $N, T, E_{K_{ab}}(N)$. Because N is independent of $\hat{B}_a$ and T is related only to $\hat{B}_a \cap \hat{B}_b$, so the only useful information can be deduced by Bob from this message is $\hat{B}_a \cap \hat{B}_b$. □

Claim 4 *If honestly following the PSP protocol, Eve cannot be successfully paired with Bob as the initiator of the protocol.*

Proof According to PSP, the sufficient and necessary condition for Eve to be successfully paired with Bob as the initiator of the protocol is that $\hat{B}_e$ is sufficiently similar to $\hat{B}_b$. In our system model (Fig. 2), Eve is far from Bob, which makes her own CSI measurements differ greatly from those taken by Bob. Therefore, using $\hat{B}_e$ generated from Eve's real CSI measurements, pairing cannot be successful. From this, if Eve wants to pass the Bob's authentication, the only possible way is to forge $\hat{B}_e$ based on $\hat{B}_b$. However, according to Claim 2, Eve cannot access $\hat{B}_b$ based on the messages received from Bob. Therefore, if honestly following the PSP protocol, Eve cannot be successfully paired with Bob as the initiator of the protocol □

Claim 5 *If honestly following the PSP protocol, Eve cannot be successfully paired with Alice as the responder of the protocol.*

Proof As described in the proof of Claim 4, Eve cannot pass Alice's authentication if $\hat{B}_e$ is generated from her real CSI measurements. According to Claim 3, Eve cannot forge $\hat{B}_e$ based on the information received from Alice else. Therefore, if honestly following the PSP protocol, Eve cannot be successfully paired with Alice as the responder of the protocol. □

Claim 6 *If honestly following the PSP protocol, only the devices located within the legitimate distance can be successfully paired.*

Proof According to Claims 4 and 5, this claim can be proved directly. □

6.1.3 All-zero-polynomial attack prevention

Remarkably, Eve can define a polynomial P_e, of which the coefficients are all zeros. In such a situation, because each $x_i \in \hat{B}_a$ ($y_i \in \hat{B}_b$) is a root of P_e, Eve can obtain B_a (B_b) by sending P_e to Alice (Bob). To prevent this all-zero-polynomial attack, when defining the polynomial, we make it mandatory to set the coefficient of its highest degree term to 1 (*i.e.*, a_n in the basic protocol and $a_{i,\upsilon}$ in the modified protocol), the same approach as in [41].

6.1.4 The quality of the key

The quality of the generated secret key depends on its length and randomness. Obviously, it is easy to control the length of the key by controlling the number of CSI samples collected by Alice and Bob. For PSP, the randomness guarantee comes from two aspects. First, the radio channel varies in unpredictable ways over short time scales due to environmental factors, which provides the inherent randomness for B_a and B_b. Second, in our method, the non-adjacent subcarriers selected for bitstream generation avoid the occurrence of repeated bit segments in B_a and B_b, which improves the entropy of the generated key. Based on the above analysis, we obtain the conclusion that a high-quality key can be generated by PSP that can effectively resist brute-force key crack attacks.

6.2 Performance analysis

6.2.1 Time complexity

Now, we discuss the time complexity of the authentication and key generation protocol (Protocol 2). In our protocol, Alice first generates a pair of keys, the time of complexity of which is $O(1)$. Then, Alice has to define the polynomials according to $\hat{B}_a$. In this step, the $\hat{B}_a$ with length n is divided into m groups, and each group contains υ elements. Therefore, its time complexity is $O(m\upsilon^2)$. Next, the coefficients of these polynomials are encrypted, and each υ-degree polynomial needs υ times encryptions. Therefore, the time complexity of this step is $O(\upsilon m) = O(n)$. After that, Bob has to compute $E_{PU_a}(rP_a(y_k) + y_k)$. For each y_k in $\hat{B}_b$, the technique needs $O(\upsilon)$ exponentiations to compute $E_{PU_a}(rP_{X_i}(y_k) + y_k)$. Therefore, the time complexity of this step is $O(\upsilon n)$. In the next step, Alice has to decrypt each ciphertext in R_{ab}. Therefore, the time complexity is $O(n)$. Finally, the time complexity of the step in which Bob implements the authentication of Alice is $O(1)$.

6.2.2 Robustness

Due to the presence of noise and manufacturing variations, the bitstreams generated by two devices that come within close proximity of each other may not be completely consistent. We use the bit error rate to describe the degree of the inconsistency, which is defined as the number of inconsistent bits over the total length of the bitstream. Our experiments show that the bit error rate will increase rapidly with increasing distance between two devices. In this paper, the high robustness means that these two devices can be successfully paired with a high probability within the legitimate distance even under the condition of higher bit error rate.

To analyze the robustness, we compare PSP to an error correction code (ECC)-based approach and a parity-check-based approach. Both ECC-based and parity-check-based approaches tend to make these two inconsistent bitstreams become identical, which is called information reconciliation. Using the ECC-based approach, both B_a and B_b are treated as distorted versions of "some" n-bit codeword of an (n, k) error correcting code. The code C to be used is known to all parties, including Eve. Let $f_c(\cdot)$ be the decoding function of C that maps any n-bit sequence to the closet codeword in C. Obviously, if B_a and B_b are very similar, $f_c(B_a)$ will be equal to $f_c(B_b)$ with high probability. Using parity-check, a number of rounds of parity checks need to be implemented between two devices. Each round check can find and correct 1 error bit. Actually, the basic principles of these two types of approaches are the same, which is error correcting. Intuitively, for any error correcting algorithm, its error correcting ability is bounded and will be deteriorated rapidly with increasing bit error rate. We conduct a simulation to validate it. We generate two identical bitstreams B_a and B_b, which have the same length of 100 bits. Then, we add different numbers of inconsistent bits into B_a and B_b (from 1 bit to 15 bits), and we use (23,12) golay code (ECC based) [7] and cascade (parity-check based) [10] for information reconciliation, respectively. For each bit error setting, we choose the positions of inconsistent bits randomly and repeat 50 experiments. Figure 8 shows the experimental results. As we can see from it, when the number of inconsistent bits exceeds 10, both the ECC-based approach and the parity-check-based approach cannot make B_a and B_b identical with a high probability. Specifically, the ECC-based and parity-check-based approaches cannot work stably when the bit error rate exceeds 10%.

Our scheme is based on a principle that is completely different from the above-mentioned methods. Instead of making B_a and B_b identical, we achieve authentication and key generation by taking advantage of $B_a \cap B_b$. The robustness of our protocol depends on an appropriate threshold value τ of $|\hat{B}_a \cap \hat{B}_b|$ chosen by Alice and Bob. If τ is set as too large, an illegitimate device may pass the authentication, which is called a *false positive* case. If τ is set as too small, this will result in a situation where two legitimate devices can only be paired within a very small distance range. Fortunately, due to the positive correlation between the distance and the bit error rate, it is not difficult to determine the value of τ. According to Fig. 9, the maximum bit error rate is approximately 50%. Let us consider the extreme situation. For the first situation, all inconsistent bits in B_a and B_b are scattered in different $\hat{B}_a[i]$ and $\hat{B}_b[i]$. Therefore, the number of consistent elements between $\hat{B}_a$ and $\hat{B}_b$ is 0. Let the length of $\hat{B}_a$ and $\hat{B}_b$ be

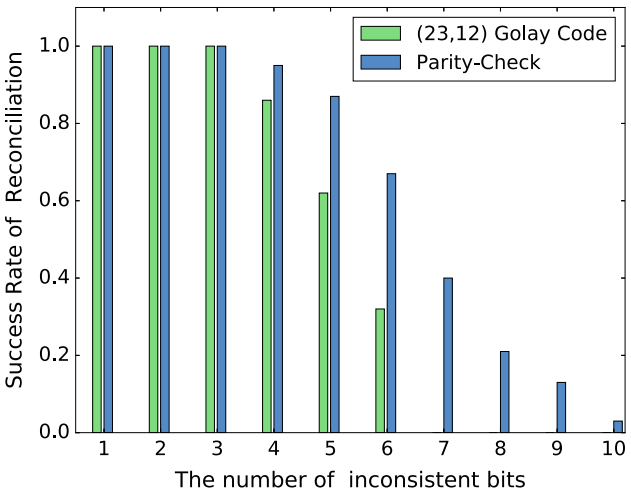

Fig. 8 Fault tolerance of ECC and parity-check

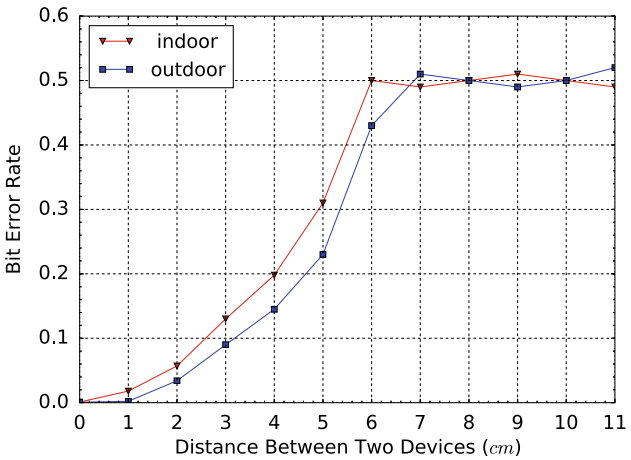

Fig. 9 Bit error rate versus distance

N. For the second situation, all inconsistent bits in B_a and B_b occur in the same $\hat{B}_a[i]$ and $\hat{B}_b[i]$. At this time, the number of consistent elements between $\hat{B}_a$ and $\hat{B}_b$ is $0.5N$. Therefore, choosing a value of τ that is greater than $0.5N$ can almost fully eliminate the false positive cases. If we set τ to be $0.6N$, Alice and Bob can be successfully paired with at least a 20% bit error rate. According to the above analysis, our approach is more robust than the ECC- and parity-check-based approaches. Our experimental results shown in Sect. 6 will further validate this.

7 Evaluation

In this section, we evaluate the performance of PSP by conducting a series of experiments and comparisons. We first investigate the relationship between the bit error rate

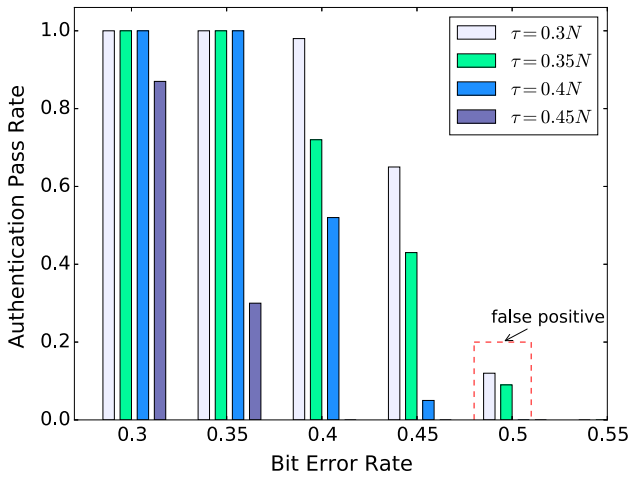

Fig. 10 The impact of the threshold τ

and the distance. Then, we show how to determine an appropriate threshold τ according to this relationship. Finally, we evaluate the performance of our approach in terms of efficiency, security and robustness.

7.1 Methodology

We conduct experiments with four laptops, named Alice, Bob, Eve and Peter. The laptops are all equipped with commodity off-the-shelf Intel 5300 wireless NICs. Peter is configured as an AP. The wireless connections among the four laptops operate in the 802.11n 2.4 GHz channel. Their NIC clocks are synchronized. As shown in Fig. 2, the antennas of Alice and Bob are located in less than 0.5λ, whereas Eve is deployed at least one λ away from Alice and Bob. As the AP, Peter broadcasts beacons every 50 ms. The prototype of our scheme is built in Python. The implementation of PSI is based on PHE 1.2.2 [42], which is a library for partially homomorphic encryption in Python. We conduct our experiments under scenarios for both indoors and outdoors. In both scenarios, all equipment remains static.

7.2 Bit error rate versus distance

As was noted in the previous section, the closer two devices become, the more similar their CSI measurements become. In other words, the closer two devices become, the more consistent their generated bitstreams are. To validate this intuition, we first conduct experiments to investigate the relationship between the bit error rate and the distance. Here, the bit error rate is defined as follows.

Definition 1 (*Bit error rate*) Assume that the total number of bits in B_a and B_b is N ($len(B_a) = len(B_b) = N$), the number of inconsistent bits between B_a and B_b is M, and the bit error rate is defined as M/N.

In our experiments, we gradually increase the distance between Alice and Bob (from 1 to 11 cm), and we record the bit error rates at different distances. The experimental results shown in Fig. 9 are in good agreement with our expectations. As we can see from the figure, when the distance between Alice and Bob is less than 1 cm, B_a and B_b are almost completely consistent. When the distance exceeds 3 cm, the bit error rate increases rapidly. The bit error rate achieves a peak value of approximately 50% when the distance exceeds 6 cm (about 0.5λ). In addition, because the outdoor multi-path is much sparser than that of indoors, at the same distance setting, the outdoor bit error rate is less than the indoor bit error rate.

7.3 The impact of the threshold τ

According to the analysis in Sect. 6, the robustness of our protocol mainly depends on the value of the threshold τ. If τ is set as too large, an illegitimate device may pass the authentication, which is called the *false positive* case. If τ is set as too small, this will result in the case where two legitimate devices can only be paired within a very small distance range. To determine the appreciate value of τ, we conduct the following experiments. Let N be the length of $\hat{B}_a$ ($len(\hat{B}_a) = len(\hat{B}_b)$), and let ε be the bit error rate between B_a and B_b. For each parameter combination (τ, ε), we conduct 50 independent experiments. In each experiment, we first randomly generate a bitstream with a length of 1000 bits, and we take this as B_a. Then, we randomly choose $1000 \times \varepsilon$ bits in B_a and change their value (change 1 to 0 and 0 to 1). The modified bitstream is taken as B_b. Then, B_a and B_b are converted to $\hat{B}_a$ and $\hat{B}_b$. Finally, taking $\hat{B}_a$, $\hat{B}_b$ and τ as inputs, we run the authentication and key generation protocol (Protocol 2). For each parameter combination (τ, ε), we record the authentication pass rate, which is defined as follows.

Definition 2 (*Authentication pass rate*) Given the fixed threshold τ and the fixed bit error rate, we perform K trials in which the PSP is run to make Alice and Bob authenticate each other. Assume that the number of the successful experiments (the authentication is passed) is R, and the authentication pass rate is defined as R/K.

Figure 10 shows the experimental results. As we can see, if $\tau = 0.3N$ or $\tau = 0.35N$, the authentication can be passed when the bit error rate achieves 50%. Specifically, according to the relationship between the bit error rate and the distance shown in Fig. 9, a device that is located beyond the legitimate distance may pass the authentication

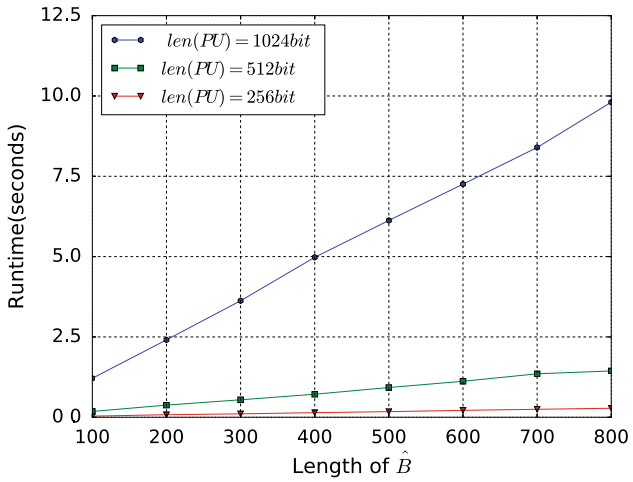

Fig. 11 The time cost of authentication

(false positive case). To eliminate false positive cases, we set τ to be $0.4N$ in our following experiments, which can maximize the authentication distance between two legitimate devices.

7.4 Efficiency evaluation of the authentication

In the simulation environment, we test the time cost of our authentication and key generation protocol (Protocol 2). We generate the sets $\hat{B}_a$ and $\hat{B}_b$ artificially, fill them with the same number of random elements, and record the time from defining the polynomial to working out the intersection by both sides. Figure 11 shows the experimental results. As we can see, the length of PU_a (homomorphic encryption key) has a great impact on efficiency. Obviously, the longer the key is, the greater the security becomes and the lower the efficiency. In the following experiments, we set the length of PU_a to be 128 bits.

7.5 Randomness of the key

Randomness is significant for measuring the security of a secret key. In our work, we use the NIST statistical Test Suite to check the randomness of keys generated by our scheme. According to the specification in this suite, a p-value is the probability of obtaining a test statistic as equal to or greater than the one observed if the sequence is random. Hence, a smaller p-value indicates that the sequence is more unlikely to be random. Passing the NIST test requires a p-value of greater than 0.01. We list the p-value of our scheme in 7 types of tests in Table 2. From the results, we can see that the generated keys pass all the test.

Table 2 NIST statistical test suite results

Test	Indoor	Outdoor
Frequency	0.712	0.535
Long run of 1 s	0.311	0.314
FFT	0.611	0.683
Approx. entropy	0.591	0.798
Cum.sums (fwd)	0.412	0.547
Cun.sums (rev)	0.553	0.574
Runs	0.721	0.582

7.6 Comparison

We compare PSP with the ECC-based and parity-check-based approaches in terms of efficiency and robustness. In our experiments, we gradually increase the distance between Alice and Bob. For each distance setting, we collect 20 sets of CSI measurements and generate the bit-streams using the method mentioned in Sect. 4. Then, we use the PSP, (23,12) Golay code and parity-check to extract the shared 128-bit symmetric key. We use the three following metrics for comparison. The *communication counts* are defined as the number of one-way communications between Alice and Bob in the key agreement process. The experimental results are shown in Fig. 12.

Definition 3 (*Bit generation rate*) Assume that the total number of secret bits of the shared key is N, the total time cost of the generated key is T, and the bit generation rate is defined as N/T.

Definition 4 (*Pairing success rate*) Assume that the total number of trials with a fixed distance setting is H, the number of trials in which the shared key can be generated successfully is S, and the pairing success rate is defined as S/H.

Definition 5 (*Communication counts*) The communication counts are defined as the number of one-way communications between Alice and Bob in the key agreement process.

As we can see from Fig. 12(a, b), when the distance is less than 0.1λ (1 cm), the bit generation rate of PSP is less than that of the (23,12) Golay code and parity-check. That is because the computation cost of the homomorphic encryption is higher. However, when the bit generation rate of the two other approaches decreases rapidly with increasing distance, PSP can still ensure a stable bit generation rate.

In addition to the comparison of the bit generation rate, the paring success rate of the (23,12) Golay code and parity-check also decrease rapidly with increasing distance

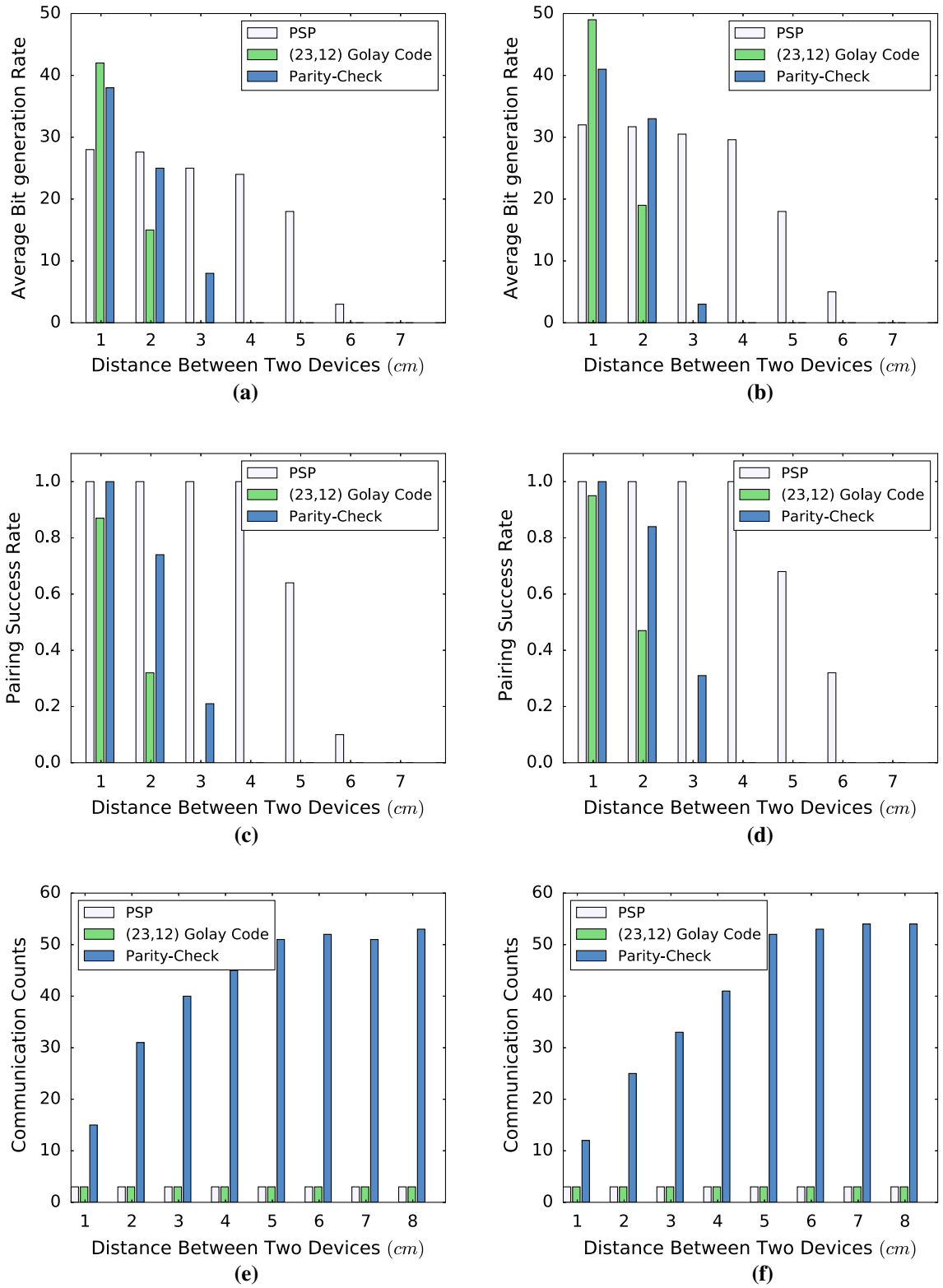

Fig. 12 Comparison. **a** Bit generation rate (indoor), **b** bit generation rate (outdoor), **c** pairing success rate (indoor), **d** pairing success rate (outdoor), **e** communication counts (indoor), **f** communication counts (outdoor)

(Fig. 12c, d), which is consistent with the analysis in Sect. 5. In contrast, PSP can successfully pair Alice and Bob with a high probability even when the distance becomes 5 cm.

Finally, as we can see from Fig. 12(e, f), the communication counts of PSP and (23,12) Golay Code remain constant (3 times). In contrast, the communication counts of parity-check increase rapidly with increasing distance. That is because, with increasing bit error rate, additional rounds of parity check are required for error correcting.

According to the comparison results, we can obtain the conclusion that PSP is more robust than the approaches based on error correcting techniques in terms of efficiency and authentication distance.

8 Conclusion

In this paper, we propose an authenticated key agreement scheme, named PSP, which can be used for proximity-based secure pairing. PSP exploits CSI measured from OFDM subcarriers as the proof of authentication and the source of security, and the technique can achieve bidirectional authentication and key generation based on homomorphic encryption and PSI. Through theoretical analysis and experimental evaluation, we show the feasibility, security, efficiency and robustness of PSP. The experimental results show, compared with existing works, that PSP has a more stable secret bit generation rate and enables a longer authentication distance while ensuring efficiency and security.

References

1. Kang, H. J., Park, K. Y., Cho, K., & Kang, C. G. (2014). Mobile caching policies for device-to-device (D2D) content delivery networking. In *Computer Communications Workshops (INFOCOM WKSHPS), 2014 IEEE Conference on* (pp. 299–304). IEEE.
2. Akhtar, R., Leng, S., Wu, F., & Memon, I. (2013). Improvement of content delivery in mobile social networks. In *Computational Problem-solving (ICCP), 2013 International Conference on*. (pp. 139–143). IEEE.
3. Das, A. K., Kumari, S., Odelu, V., Li, X., Wu, F., & Huang, X. (2016). Provably secure user authentication and key agreement scheme for wireless sensor networks. *Security and Communication Networks*, *9*(16), 3670–3687.
4. McCune, J. M., Perrig, A., & Reiter, M. K. (2005). Seeing-is-believing: Using camera phones for human-verifiable authentication. In *2005 IEEE Symposium on Security and Privacy (S&P'05)* (pp. 110–124). IEEE.
5. Goodrich, M. T., Sirivianos, M., Solis, J., Tsudik, G., & Uzun, E. (2006). Loud and clear: Human-verifiable authentication based on audio. In *26th IEEE International Conference on Distributed Computing Systems (ICDCS'06)* (p. 10). IEEE.
6. Mayrhofer, R., & Gellersen, H. (2007). Shake well before use: Authentication based on accelerometer data. In *International Conference on Pervasive Computing* (pp. 144–161). Springer.
7. Mathur, S., Miller, R., Varshavsky, A., Trappe, W., & Mandayam, N. (2011). Proximate: Proximity-based secure pairing using ambient wireless signals. In *Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services* (pp. 211–224). ACM.
8. Varshavsky, A., Scannell, A., LaMarca, A., & De Lara, E. (2007). Amigo: Proximity-based authentication of mobile devices. In *International Conference on Ubiquitous Computing* (pp. 253–270). Springer.
9. Rappaport, T. S. (1996). *Wireless communications: Principles and practice* (Vol. 2). New Jersey: Prentice Hall PTR.
10. Jana, S., Premnath, S. N., Clark, M., Kasera, S. K., Patwari, N., & Krishnamurthy, S. V. (2009). On the effectiveness of secret key extraction from wireless signal strength in real environments. In *Proceedings of the 15th Annual International Conference on Mobile Computing and Networking* (pp. 321–332). ACM.
11. Premnath, S. N., Jana, S., Croft, J., Gowda, P. L., Clark, M., Kasera, S. K., et al. (2013). Secret key extraction from wireless signal strength in real environments. *IEEE Transactions on Mobile Computing*, *12*(5), 917–930.
12. Liu, H., Yang, J., Wang, Y., & Chen, Y. (2012). Collaborative secret key extraction leveraging received signal strength in mobile wireless networks. In *INFOCOM, 2012 Proceedings IEEE* (pp. 927–935). IEEE.
13. Zan, B., Gruteser, M., & Hu, F. (2012). Improving robustness of key extraction from wireless channels with differential techniques. In *2012 International Conference on Computing, Networking and Communications (ICNC)* (pp. 980–984). IEEE.
14. Liu, H., Wang, Y., Yang, J., & Chen, Y. (2013). Fast and practical secret key extraction by exploiting channel response. In *INFOCOM, 2013 Proceedings IEEE* (pp. 3048–3056). IEEE.
15. Xi, W., Li, X. Y., Qian, C., Han, J., Tang, S., Zhao, J., et al. (2014). Keep: Fast secret key extraction protocol for D2D communication. In *2014 IEEE 22nd International Symposium of Quality of Service (IWQoS)* (pp. 350–359). IEEE.
16. Liu, Y., Draper, S. C., & Sayeed, A. M. (2012). Exploiting channel diversity in secret key generation from multipath fading randomness. *IEEE Transactions on Information Forensics and Security*, *7*(5), 1484–1497.
17. Perahia, E., & Stacey, R. (2013). *Next generation wireless LANS: 802.11 n and 802.11 ac*. Cambridge: Cambridge University Press.
18. Renner, R., & Wolf, S. (2005). Simple and tight bounds for information reconciliation and privacy amplification. In *International Conference on the Theory and Application of Cryptology and Information Security* (pp. 199–216). Springer.
19. Cachin, C., & Maurer, U. M. (1997). Linking information reconciliation and privacy amplification. *Journal of Cryptology*, *10*(2), 97–110.
20. Brassard, G., & Salvail, L. (1993). Secret-key reconciliation by public discussion. In *Workshop on the Theory and Application of of Cryptographic Techniques* (pp. 410–423). Springer.
21. Arain, Q., Zhongliang, D., Memon, I., et al. (2016). Privacy preserving dynamic pseudonym-based multiple mix-zones authentication protocol over road networks. *Wireless Personal Communications*, *95*, 1–17.
22. Memon, I., Arain, Q. A., Memon, H., et al. (2017). Efficient user based authentication protocol for location based services discovery over road networks. *Wireless Personal Communications*, *95*, 1–20.
23. Memon, I., Mohammed, M. R., Akhtar, R., et al. (2014). Design and implementation to authentication over a GSM System using certificate-less public key cryptography (CL-PKC). *Wireless Personal Communications*, *79*, 661–686.

24. Memon, I., Hussain, I., Akhtar, R., et al. (2015). Enhanced privacy and authentication: An efficient and secure anonymous communication for location based service using asymmetric cryptography scheme. *Wireless Personal Communications*, *84*, 1487C–1508.
25. Kamenyi, D. M., Wang, Y., Zhang, F., Memon, I., & Gustav, Y. H. (2013). Authenticated privacy preserving for continuous query in location based services. *Journal of Computational Information Systems*, *9*(24), 9857–9864.
26. Gustav, Y. H., Wang, Y., Domenic, M. K., Zhang, F., & Memon, I. (2013). Velocity similarity anonymization for continuous query location based services. In *Computational Problem-solving (ICCP), 2013 International Conference on* (pp. 433–436). IEEE.
27. Memon, I., & Arain, Q. A. (2016). Dynamic path privacy protection framework for continuous query service over road networks. *World Wide Web*, *20*(4), 639–672.
28. Maurer, U. M. (1993). Secret key agreement by public discussion from common information. *IEEE Transactions on Information Theory*, *39*(3), 733–742.
29. Ahlswede, R., & Csiszar, I. (1998). Common randomness in information theory and cryptography. II. CR capacity. *IEEE Transactions on Information Theory*, *44*(1), 225–240.
30. Sayeed, A., & Perrig, A. (2008). Secure wireless communications: Secret keys through multipath. In *2008 IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 3013–3016).
31. Wilson, R., Tse, D., & Scholtz, R. A. (2007). Channel identification: Secret sharing using reciprocity in ultrawideband channels. *IEEE Transactions on Information Forensics and Security*, *2*(3), 364–375.
32. Wang, Q., Su, H., Ren, K., & Kim, K. (2011). Fast and scalable secret key generation exploiting channel phase randomness in wireless networks. In *INFOCOM, 2011 Proceedings IEEE* (pp. 1422–1430).
33. Tope, M. A., & McEachen, J. C. (2001). Unconditionally secure communications over fading channels. In *Military Communications Conference, 2001. MILCOM 2001. Communications for Network-Centric Operations: Creating the Information Force* (Vol. 1, pp. 54–58). IEEE.
34. Mathur, S., Trappe, W., Mandayam, N., Ye, C., & Reznik, A. (2008). Radio-telepathy: Extracting a secret key from an unauthenticated wireless channel. In *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking* (pp. 128–139). ACM.
35. Chou, T. H., Draper, S. C., & Sayeed, A. M. (2010). Impact of channel sparsity and correlated eavesdropping on secret key generation from multipath channel randomness. In *2010 IEEE International Symposium on Information Theory* (pp. 2518–2522).
36. Halperin, D., Hu, W., Sheth, A., & Wetherall, D. (2011). Tool release: Gathering 802.11 n traces with channel state information. *ACM SIGCOMM Computer Communication Review*, *41*(1), 53.
37. Okamoto, T., & Uchiyama, S. (1998). A new public-key cryptosystem as secure as factoring. In *International Conference on the Theory and Applications of Cryptographic Techniques* (pp. 308–318). Springer.
38. Naccache, D., & Stern, J. (1998). A new public key cryptosystem based on higher residues. In *Proceedings of the 5th ACM Conference on Computer and Communications Security* (pp. 59–66). ACM.
39. Paillier, P. (1999). Public-key cryptosystems based on composite degree residuosity classes. In *International Conference on the Theory and Applications of Cryptographic Techniques* (pp. 223–238). Springer.
40. Freedman, M. J., Nissim, K., & Pinkas, B. (2004). Efficient private matching and set intersection. In *International Conference on the Theory and Applications of Cryptographic Techniques* (pp. 1–19). Springer.
41. Li, R., & Wu, C. (2007). An unconditionally secure protocol for multi-party set intersection. In *Applied Cryptography and Network Security* (pp. 226–236). Springer.
42. GitHub Inc. (2016). A library for partially homomorphic encryption in Python. https://github.com/NICTA/python-paillier. Accessed.

Weirong Cui is a Ph.D. candidate in the Department of Computer Science at the Northwestern Polytechnical University, Xi'an, China. He received his M.S. degree in computer science from Huazhong University of Science and Technology, Wu Han, China, in 2008. His research and teaching interests are in network security and privacy protection in Mobile Social Networks.

Chenglie Du is a professor in the Department of Computer Science at the Northwestern Polytechnical University, China. He received his Ph.D. degree in computer science from the same institution in 1999. His research and teaching interests are in scheduling theory, real-time distributed computing systems, design and verification of Cyber-Physical Systems, and domain software engineering.

Jinchao Chen is a Ph.D. candidate in the Department of Computer Science at the Northwestern Polytechnical University, Xi'an, China. He received his M.S. degree in computer science from the same institution in 2012. His research interests are in multiprocessor scheduling theory, real-time systems design, and ubiquitous computing.