Key Establishment for Secure Asymmetric Cross-Technology Communication

Wei Wang
wei.wang.5@slu.edu
Saint Louis University
USA

Xin Liu
liu.10663@osu.edu
The Ohio State University
USA

Zicheng Chi
z.chi@csuohio.edu
Cleveland State University
USA

Stuart Ray
stuart.ray@slu.edu
Saint Louis University
USA

Ting Zhu
zhu.3445@osu.edu
The Ohio State University
USA

ABSTRACT

Recent advances in cross-technology communication can support direct communication among heterogeneous IoT devices (i.e., WiFi, ZigBee, and BLE) without requiring any modifications to the hardware, which has significantly improved the communication efficiency and shown great advantages for supporting smart applications. However, until now a key establishment protocol to support secure and reliable asymmetric cross- technology communication (CTC) is missing, which introduces severe privacy and security issues. Existing solutions are not designed for CTC, since they mainly focus on the symmetric communication among homogeneous IoT devices. In this work, we present a Key Establishment Protocol (KEP), which explores and lever- ages the unique feature of CTC - Possibility PN Sequence Reception (PSR) to not only perform key establishment between heterogeneous IoT devices with different physical layers (i.e., WiFi and ZigBee) but also improve the communication reliability at the same time. Our extensive real-world experiments show that KEP can finish the key establishment in seconds and effectively defend against multiple types of attacks. Furthermore, KEP doubles the packet reception ratio compared to the state-of-the-art solutions.

CCS CONCEPTS

• **Security and privacy → Mobile and wireless security**.

KEYWORDS

Cross-Technology Communication Security; IoT Security

ACM Reference Format:
Wei Wang, Xin Liu, Zicheng Chi, Stuart Ray, and Ting Zhu. 2024. Key Establishment for Secure Asymmetric Cross-Technology Communication. In *Proceedings of the 19th ACM ASIA Conference on Computer and Communications Security, July 1–15, 2024, Singapore.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3460120.3484766

ASIA CCS '24, July 1–5, 2024, Singapore, Singapore

ACM ISBN 979-8-4007-0482-6/24/07…
https://doi.org/10.1145/3634737.3637670

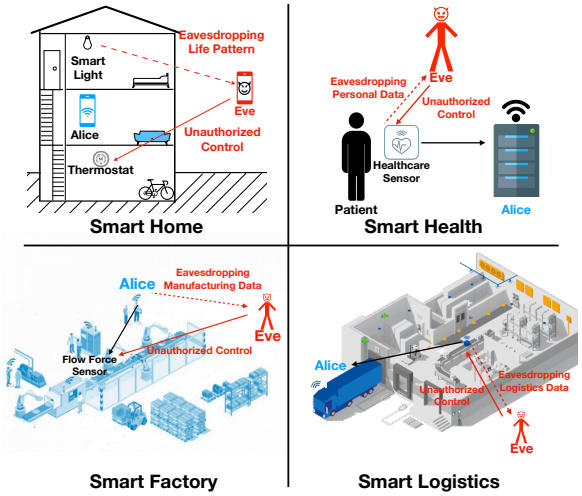


Figure 1: Due to the lack of security protocol for CTC, Eve can easily eavesdrop the cross-technology communication between WiFi and ZigBee. Moreover, Eve may also take over the control of IoT devices.

1 INTRODUCTION

With the exponential growth of Internet of Things (IoT), a huge number of smart devices are crowded in the same Industrial Scientific Medical (ISM) Band, including WiFi, Bluetooth and ZigBee [1]. On one hand, it is a common belief that the coexistence of these IoT devices reduces the spectrum utilization efficiency [32]. On the other hand, the coexistence of IoT devices also brings new opportunities for these devices to collaborate with each other [21, 22]. This double-sided recognition leads to the emerging of Cross-technology Communication (CTC) techniques which supports direct communication among heterogeneous IoT devices without a gateway. Currently, researches have shown that CTC can provide many benefits and support lots of smart applications. For example, CTC can significantly reduce the energy consumption [21, 36], provide efficient channel coordination [35], and reduce the network delay [32] for IoT devices. Furthermore, with the help of CTC techniques, the occupants in a smart building or smart home can directly use WiFi to control smart ZigBee devices, such as smart light, smart thermostat, and other IoT devices [11, 22].

To enable WiFi and ZigBee communication, recent advances in cross-technology communication (CTC) mainly utilize WiFi signals

to emulate ZigBee waveforms (i.e., WEBee [22]). To do this, the WiFi device controls the payload of a WiFi frame so that a portion of the frame can be recognized by commodity ZigBee devices transparently as a legitimate ZigBee frame. This technique can achieve high throughput and long distance communication from WiFi to ZigBee without any hardware modifications. However, a fundamental problem that limits the applications for WiFi and ZigBee CTC is the vulnerability to various attacks, such as eavesdropping, spoofing, etc. Specifically, due to the lack of cross-technology communication security protocols, an attacker (Eve) can easily eavesdrop the communication between legitimate WiFi (Alice) and ZigBee devices (Bob) and even take control of legitimate ZigBee devices, which introduces severe privacy leakage and security issues. For example, as shown in Figure 1, when occupants control the smart lighting system, smart thermostat, or other smart devices in a smart home, sensitive information including occupants behavior data is exposed to Eve who is located outside the smart home. Eve can even control occupants' thermostat to change the temperature of the house, which can be fatal to the elderly [7, 8]. Similarly, by leveraging CTC techniques, it is easier for the patient to directly update their health data to the server. However, Eve may also get the personal heath data of the patient and control the health sensor at the same time. In smart factory or smart logistics scenarios, the coexistence of heterogeneous IoT devices (i.e., WiFi and ZigBee) requires these devices collaborate with each other to improve the manufacturing efficiency. For instance, by using CTC techniques, the WiFi server (Alice) can directly communicate with the flow force sensor to control the machine tools in a smart factory while the packages with embedded ZigBee sensors can directly update the warehouse storage information to Alice for efficient delivery. However, due to the lack of security protocols, Eve can easily eavesdrop the sensitive information transmitted through these sensors and further control the entire factory. Therefore, to protect legitimate communication, a key establishment process among heterogeneous IoT devices (i.e., WiFi and ZigBee) is the critical first step to secure the cross-technology communication channel.

However, due to the limitations of the physical layer, current key establishment approaches cannot be utilized for asymmetric CTC (i.e., CTC between WiFi and ZigBee). Specifically, for pre-shared key-based (PSA) approaches, the pre-shared secret key may be lost or leaked. In addition, PSA-based approaches normally suffer scalability issues, which is not suitable for heterogeneous IoT networks [17, 29]. Conventional public key cryptography approaches (e.g., Diffie–Hellman) [15, 27] is considered more secure, which requires the sender and receiver to accurately exchange the secret information. However, in CTC, due to the limitations of the 802.11 physical layer, the emulated ZigBee signal cannot perfectly match the desired ZigBee signal. In the worst case, some specific emulated ZigBee signals will never be recognized by the ZigBee receiver, which introduces *inevitable* errors and results in key exchange failure. On the contrary, physical layer-based key establishment approaches are secure and scalable. They mainly utilize the Received Signal Strength (RSS), the Channel Impulse Response (CIR), and the Channel State Information (CSI) to extract secret bits. Specifically, these approaches require Alice and Bob to measure the variations of the wireless channel by sending the same probe signal to each other [25, 31, 33]. According to the reciprocity theory [4, 25], the

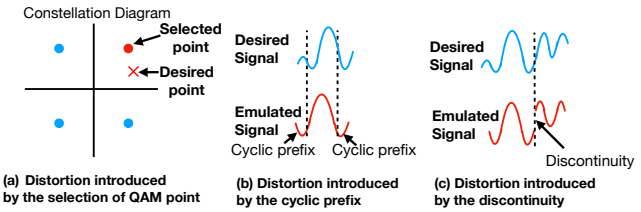


Figure 2: Distortions in the signal emulation process.

measurements of the wireless channel from Alice and Bob are expected to be similar and can be used for key establishment. However, in CTC, since Alice (WiFi) and Bob (ZigBee) have different physical layers, they cannot transmit the same probe signal to each other, which makes existing approaches useless for key establishment. Therefore, to conduct key establishment, the first challenge is **how to extract and measure the features of the asymmetric channel between WiFi and ZigBee for key establishment**. Moreover, we need to answer the question of **how to convert these features into secret bits**. Besides these two challenges, it is also important to **improve the communication reliability for WiFi-to-ZigBee CTC**.

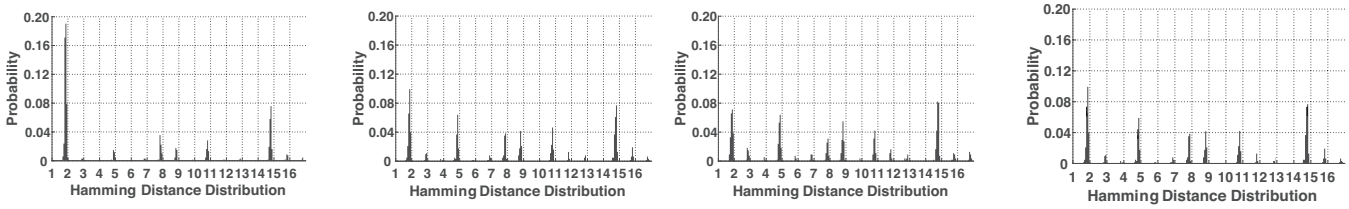
To overcome the above challenges, we introduce a novel security protocol called KEP that can perform key establishment and improve communication reliability for WiFi and ZigBee asymmetric CTC simultaneously. The key idea of KEP is to leverage the imperfect emulated ZigBee signal for key establishment. Specifically, by deeply exploring CTC techniques and conducting real world experiments, we found that the imperfect emulation in ZigBee signals and the randomness of the wireless channel increases the demodulation uncertainty at the ZigBee receiver side. As a result, even if the WiFi broadcasts the same emulated ZigBee signal, the PN sequences received by ZigBee devices will be different. In this paper, we name this unique feature as the **Possibility PN Sequence Reception (PSR)**. Since the imperfect emulation is determined by the fixed physical layer while the wireless channel varies from location to location, PSR is ***location-specific*** and can be utilized for secret key establishment. Moreover, by analyzing the PSR, WiFi can compensate the WiFi-to-ZigBee CTC and improve the communication reliability at the same time.

We summarize the contributions of KEP as follows:

- To the best of our knowledge, KEP is the first work that focuses on the security and reliability of the asymmetric CTC between WiFi and ZigBee at the same time. It fills the gap of secure CTC and opens a promising direction for future secure CTC techniques.
- To the best of our knowledge, KEP is the first work that in-depth explores WiFi-to-ZigBee CTC and discovers an unique feature, PSR, in CTC. It also answers the question of how to leverage PSR to perform key establishment and improve the communication reliability at the same time.
- We extensively evaluate our design under various real-world settings. The evaluation results show that KEP can finish the key establishment in seconds and double the WiFi to ZigBee packet reception ratio.

2 OBSERVATION AND SYSTEM MODEL

In this section, we first analyze and show the observation of Possibility PN Sequence Reception (PSR). Then, we define the System and Security Model of KEP.



(a) Decoded PN sequences at Bob's side (outdoor scenario).
(b) Decoded PN sequences at Bob's side (indoor scenario).
(c) Decoded PN sequences at Calvin's side (indoor scenario).
(d) Decoded PN sequences at Dave's side (indoor scenario)

Figure 3: The hamming distance distribution when WiFi transmits the emulated ZigBee PN sequence 1.

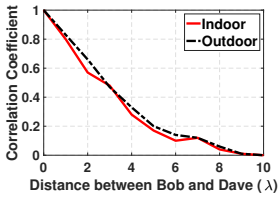


Figure 4: Correlation coefficient with the increasing of distance between Bob and Dave.

2.1 Analysis and Observation

Normally, to generate the emulated ZigBee signal, the WiFi device controls its payload so that the transmitted RF signal is similar to the ZigBee signal. However, as illustrated in Figure 2, due to the limitation of the 802.11 physical layer, the emulated signal is not exactly the same as the desired ZigBee signal for the following reasons: **First,** WiFi should select the nearest QAM constellation point according to the desired ZigBee signal. The Minimum Euclidean Distance between the selected QAM point and the desired point introduces distortions. **Second,** the WiFi uses the Cyclic Prefix (CP) to eliminate the Inter-Symbol Interference (ISI) and the Inter-Carrier Interference (ICI) while the ZigBee signal does not have cyclic prefix, which introduces imperfect emulation. **Third,** the duration of one WiFi symbol is $4\mu s$ while the duration of one ZigBee symbol is $16\mu s$. Therefore, a WiFi device needs to use four WiFi symbols to emulate one ZigBee symbol. The discontinuity between each WiFi symbol also introduces distortions.

For the ZigBee device, it uses a 32 Pseudo-random Noise Chip Sequence (PN Sequence) to express a 4-bit symbol for chip error tolerance. This technique is also known as the Direct Spreading Spectrum Sequence (DSSS). In practice, although hardware defects, multipath effects and the imperfect wireless environments (i.e., noise and interference) introduce distortions to ZigBee signals, as long as the number of chip errors is lower than the threshold, the ZigBee communication still remains reliable. However, in CTC, the imperfect emulation introduces additional distortion, which makes it challenging for the ZigBee device to detect and receive the emulated ZigBee signal. In this work, we found that the errors of the received ZigBee PN sequence vary from location to location. In fact, these errors reveal the channel conditions and can be used for key establishment.

To prove our analysis, we conduct experiments in both outdoor (line-of-sight) and indoor (non-line-of-sight) scenarios. We use USRP B210 as the WiFi device (Alice) and utilize the physical layer emulation technique [22] to conduct WiFi-to-ZigBee CTC. The WiFi is set to emulate and transmit all 16 ZigBee PN sequences and each transmission is repeated for 1×10^4 times. Since these experiments show similar trends, we show the results of the actual received (decoded) PN sequences by ZigBee (Bob) when WiFi is emulating the ZigBee PN sequence 1. Figure 3a and Figure 3b show the hamming distance distribution between the actual received PN sequence and all 16 ZigBee PN sequences. As shown in these figures, the distributions are significantly different. For the outdoor scenario, around 40% of decoded PN sequences are recognized as the ZigBee PN sequence 1 and their corresponding hamming distance are distributed between ZigBee PN sequence 1 and 2. However, for the indoor scenario, around 18% and 15% of the decoded PN sequences are recognized as the ZigBee PN sequence 14 and PN sequence 4, respectively. This experiment proves our analysis of the imperfect emulation. Moreover, we also can observe that the distributions of the actual received PN sequences vary according to different scenarios. Based on above analysis and observation, we define this unique feature of CTC as Possibility PN Sequence Reception:

Possibility PN Sequence Reception (PSR). *For CTC physical layer emulation techniques, the Possibility PN Sequence Reception represents the fact that ZigBee devices at different locations will receive different PN Sequences even if the WiFi is transmitting the same emulated signals.*

2.2 Feasibility of PSR

To use PSR for key extraction, it should satisfy the uniqueness requirement. In other words, devices at different locations should show distinct PSR. Moreover, the PSR measured from the same device at the same location should be relatively stable.

As shown in Figures 3c and 3d, we first conduct the experiments to study the uniqueness of PSR. In these experiments, Alice continuously broadcast emulated signals to Bob, Calvin and Dave in the indoor scenario. Dave is placed 0.1 meter away from Bob while Calvin is placed 3 meters away from Bob. In Figure 3c, we can observe that the distribution of the actual received PN Sequences for Calvin is different from that for Bob in Figure 3b. This is because the emulated signals are affected by the conditions of the wireless channel, such as noise, interference, multipath effect, shadowing, etc. Different from traditional communication systems (i.e., the sender and the receiver have the same physical layer), since emulated signals cannot perfectly match the desired ZigBee signal in CTC, a small change of the wireless environment will affect the received PN sequences. On the contrary, as shown in Figure 3d, the distribution of the received PN Sequences for Dave is almost the same as that for Bob in Figure 3b.

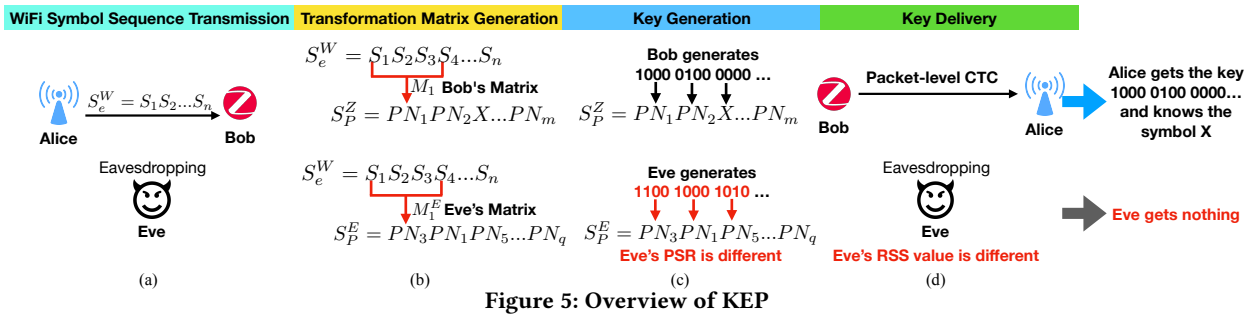


Figure 5: Overview of KEP

In Figure 4, we also study the Pearson correlation coefficient to learn the minimal distance for getting uncorrelated PSR. In this experiment, the distances between Bob and Dave vary from 0.1λ to 10λ ($\lambda = 12.5cm$ for 2.4 GHz). As we can observe from this figure, when the distance is higher than 2λ, the correlation coefficient between Bob and Dave is relatively low. As the distance increases, the correlation quickly drops to 0. Therefore, we can conclude that the PSR detected by two devices is uncorrelated as long as the distance between these devices are higher than 2λ.

In summary, PSR has following important properties for key establishment:

- *For multiple ZigBee devices, their PSRs are similar if and only if these devices are very close to each other.*
- *For a ZigBee device, the change of the location will result in the change of PSR.*

These properties are important to prove that the PSR can be used as the unique profile for key establishment.

2.3 Security Model

KEP solves the fundamental problem in a CTC system: The WiFi and ZigBee devices should perform key establishment before conducting transmission, which is the first step to build a secure CTC channel. Formally, we assume a legitimate WiFi (Alice) and ZigBee (Bob) directly communicate with each other using CTC techniques. Alice can transmit packets to Bob using a WiFi-to-ZigBee (W2Z) physical layer emulation CTC while Bob is using a ZigBee-to-WiFi (Z2W) CTC for ZigBee to WiFi communication. They have no prior shared secret. To perform the secret key establishment, Alice will transmit predefined emulated ZigBee signals to Bob. Due to the imperfect emulation and dynamic wireless channel conditions, the emulated signals received by Bob will be distorted. Bob can calculate the corresponding PSR based on the actual received PN sequences. The PSR will be used for channel measurement and secret bits generation. Moreover, due to the imperfect emulation, KEP should also satisfy the **Communication Reliability** requirement: it should improve the packet reception ratio for WiFi-to-ZigBee CTC.

Security Model. We consider an attacker Eve who is interested in stealing private information or conducting unauthorized operations by eavesdropping or conducting CTC to legitimate WiFi or ZigBee devices. Eve can either be WiFi or ZigBee or a device with WiFi and ZigBee radios. Eve can be static or mobile to attack WiFi and ZigBee devices. We assume that Eve is located beyond a safe distance ($2\lambda = 25cm$) to Alice or Bob to remain stealthy. We believe this is a reasonable assumption for the following two reasons: i) the legitimate user can easily detect co-located Eve; and ii) Eve may not be able to access the location of legitimate devices. For example, in a smart home scenario, the user can use WiFi to control ZigBee applications while the attacker does not have access to the user's house, where the legitimate devices are located. We also assume that Eve has complete knowledge of the proposed method. Specifically, we consider the following attack scenarios:

- *Eavesdropping*: Eve attempts to eavesdrop every transmission between WiFi and ZigBee. This attack is stealthy since Eve only passively senses the CTC channel between WiFi and ZigBee devices.
- *Spoofing Attack*: Eve tries to impersonate legitimate WiFi or ZigBee devices. It can either transmit unauthorized WiFi symbols to ZigBee using a physical layer emulation technique or communicate with ZigBee using packet-level CTC.
- *Predictable Channel Attack*: Eve tries to introduce predictable changes to the wireless channel by making planned movements between Alice and Bob. By doing this, Eve can vary the distribution of PSR and predict the actual received PN Sequence at Bob's side.

3 PROTOCOL OVERVIEW

1) WiFi Symbol Sequence Transmission. As shown in Figure 5 (a), to perform key establishment between Alice and Bob, Alice first transmits a predefined WiFi Symbol Sequence S_e^W that contains n WiFi symbols. This symbol sequence contains all possible WiFi symbols that can be used to emulate ZigBee signals, which is publicly known by Alice, Bob and Eve. Both the legitimate ZigBee device Bob and the attacker Eve can receive S_e^W and conduct demodulation to get the ZigBee PN sequences (in Section 4.1).

2) Transformation Matrix Generation. As shown in Figure 5 (b), we denote the actual received PN Sequence and the set of the entire actual received PN Sequences as PN_i and S_P^Z, respectively. Specifically, after receiving S_P^Z, Bob compares the PN_i with the predefined WiFi symbol sequence S_e^W to calculate the PSR. Since four WiFi symbols construct one ZigBee PN sequence, the total number of actual received PN sequences (m) will satisfy $m \leq \frac{n}{4}$. Then, according to PSR, Bob generates the multiple *Transformation Matrices*, which maps S_e^W to S_P^Z. For example, in Figure 5 (b), the transformation matrix M_1 maps the first four WiFi symbols (S_1, S_2, S_3 and S_4) to the PN sequence PN_1 while the total number of different transformation matrices is equal to the number of actual received PN sequences m. On the contrary, since Eve is located at a different place, its demodulation results are different from those of Bob. In this example, Eve receives $S_P^E = PN_3PN_1PN_5...PN_q$ and the corresponding transformation matrix for the first four WiFi symbols is M_1^E (in Section 4.2).

3) Key Generation. In KEP, the legitimate ZigBee device (Bob) is in charge of key generation. The bit string for the secret key

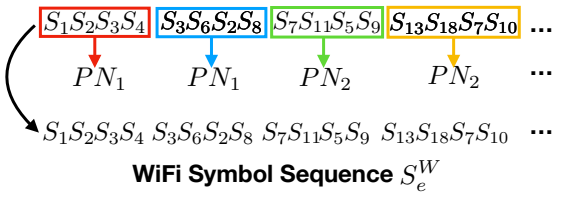


Figure 6: An example of WiFi Symbol Sequence S_e^W

should be *Suitably Long* and *Statistically Random*. Specifically, Bob constructs the secret bits according to the transformation matrix M. First, Bob will calculate the eigenvalues of each transformation matrix M. Then, these eigenvalues will be normalized and converted into secret bits using a quantizer. We also need to note that due to the imperfection emulation and the dynamic wireless channel conditions, the number of chip errors for some emulated ZigBee PN sequence may exceed the chip tolerance threshold, which cannot be recognized by Bob. To overcome this challenge and generate enough number of transformation matrices, we use a predefined bit sequence to indicate the demodulation failure. A simple example is illustrated in Figure 5 (c). The secret bits generated by Bob are 1000 and 0100 for PN_1 and PN_2, respectively. However, due to chip errors, the PN sequence from S_9 to S_{12} cannot be demodulated (denoted as X in the figure). In this case, KEP uses 0000 to indicate the demodulation failure. In contrast, since Eve's transformation matrix is different, it gets 1100, 1000 and 1010 based on M_1^E. (in Section 4.3.2).

4) Key Delivery. In Figure 5 (d), to deliver the key to Alice, Bob informs Alice of the the generated key using Z2W CTC techniques [13, 19]. Specifically, Bob controls its transmission power to indicate the generated secret bits while Alice senses its Channel State Information (CSI) or Received Signal Strength (RSS) to demodulate the signal. In contrast, since Eve's channel is independent from the channel between Alice and Bob, the detected CSI or RSS values are different and useless. (in Section 4.3.3)

5) Communication Compensation. At last, as shown in Figure 5 (c), the symbols from S_9 to S_{12} in S_e^W are not successfully demodulated, which means the corresponding WiFi symbols cannot be utilized for WiFi to ZigBee communication. In this case, to improve communication reliability, Alice will use other WiFi symbols that have been successfully demodulated as ZigBee PN sequences to represent this ZigBee PN sequence X. (in Section 4.4)

4 DETAILED PROTOCOL

In this section, we first answer the challenge of how to extract and measure the features of the wireless channel between WiFi and ZigBee devices in sections 4.1 and 4.2. Then, in section 4.3, we show how to convert these features into secret bits. At last, we show how to improve communication reliability in section 4.4.

4.1 WiFi Symbol Sequence Transmission

To perform key establishment, the first step is to transmit a predefined WiFi Symbol Sequence S_e^W from Alice (WiFi). This sequence will be used for PSR calculation by Bob (ZigBee). Specifically, S_e^W should satisfy the following requirements: *i)* The number of symbols in S_e^W should be large enough for Bob to conduct PSR analysis and generate a sufficiently long secret key; *ii)* The WiFi symbols in S_e^W should emulate all desired 16 ZigBee PN sequences so as to know which ZigBee PN sequence cannot be demodulated; and *iii)* Each ZigBee PN sequence should be emulated by different combinations of WiFi symbols in order to improve the uncertainty at Eve's side. For example, as shown in Figure 6, PN sequence PN_1 can be emulated by two different WiFi symbol combinations $S_1S_2S_3S_4$ and $S_3S_6S_2S_8$ while $S_7S_{11}S_5S_9$ and $S_{13}S_{18}S_7S_{10}$ are used to emulate PN sequence PN_2. Therefore, the WiFi symbol sequence S_e^W should contain all these WiFi symbol combinations. In practice, since there are 16 desired ZigBee PN sequences [2], the combinations of WiFi symbols in S_e^W should be able to emulate all these PN sequences.

The selections of the WiFi symbols are challenging. Specifically, a ZigBee channel is overlapped with 7 WiFi subcarriers. Therefore, for a 64-QAM WiFi, there are 64^7 (2^{42}) different WiFi symbol combinations for a $4\mu s$ WiFi transmission. Since a ZigBee PN sequence consists of four WiFi symbols, the total number of WiFi symbol combinations is $(2^{42})^4 = 2^{168}$. According to the dynamic properties of the wireless channel, some of the WiFi symbol combinations will never be recognized as the ZigBee signal by Bob. However, in practice, due to the large number of WiFi symbol combinations, it is difficult for the WiFi device to find out the suitable WiFi Symbol Sequence S_e^W.

To overcome this challenge, in KEP, the WiFi device (Alice) only needs to select a limited number of combinations. The selections are based on the Euclid Distance between the desired ZigBee signal (desired ZigBee PN sequence) and the actual WiFi QAM points. Specifically, as shown in Figure 7 (a), the WiFi device will find out the nearest QAM point with minimal Euclid Distance according to the desired ZigBee signal. This point is the optimal point for ZigBee signal emulation. Since ZigBee (Bob) uses a 32-bit PN chip sequence for chip error tolerance, some of the WiFi QAM points near the optimal point can also be used to emulate the ZigBee signal. In KEP, the points within the Euclid Distance τ of the nearest QAM point are also selected for ZigBee signal emulation, which is shown in Figure 7 (b). As a result, the WiFi only needs to transmit a limited number of combinations of WiFi symbols to the ZigBee receiver while still amplifies the uncertainty at Eve's side. Formally, the total number of combinations can be estimated as $(\frac{A\pi\tau^2}{S})^{28}$, where A is the total number of the QAM points in the constellation diagram while S is the area of the constellation diagram. In practice, A and S can be determined by the actual WiFi QAM formats. The determination of the Euclid Distance τ is more tricky. If τ is too small, it will be hard for the ZigBee device to analyze the PSR. If τ is too big, multiple emulated signals will not be recognized by the ZigBee device. In our evaluation, for a 64-QAM WiFi, τ is set to 4 to conduct efficient key establishment.

4.2 Transformation Matrix Generation

In this section, we first introduce the Chips Extension to model the relationship between the desired PN sequence and the actual received PN sequence. Then, we introduce the detailed Transformation Matrix Generation process.

4.2.1 ***Chips Extension.*** After receiving the WiFi symbol sequence S_e^W from Alice, Bod should generate the transformation matrix M. In KEP, the transformation matrix M is generated according to the relationship between the desired PN sequences and the actual

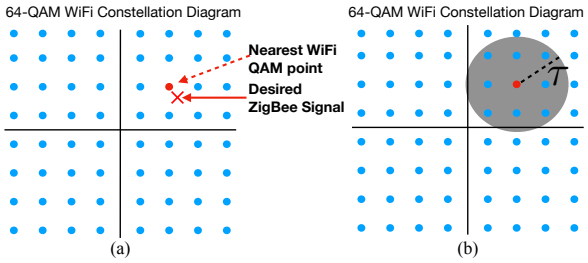


Figure 7: (a) The WiFi device finds out the nearest QAM point. (b) The QAM points within τ are selected for signal emulation.

received PN sequences. Formally, we define the PN sequences that WiFi (Alice) wants to emulate as the **desired PN sequence** S_D^Z while the **actual received PN sequence** by the legitimate ZigBee device (Bob) is denoted as S_P^Z. Then, M is calculated according to the differences between S_D^Z and S_P^Z. As mentioned in section 2.1, due to the imperfect emulation and the dynamic wireless channel conditions, there will be mismatches between S_D^Z and S_P^Z. Since the imperfect emulation is determined by the fixed 802.11 physical layer, the corresponding mismatches can be predicted while the mismatches introduced by the imperfect wireless channel conditions are hard to predict. Therefore, by calculating the transformation matrix, we can measure the randomness in the wireless channel for key establishment.

For the ZigBee device Bob, the WiFi Symbol Sequence S_e^W and its corresponding desired ZigBee PN sequence S_D^W are publicly known. Therefore, after receiving the transmissions from Alice, Bob can easily compare S_D^W with its actual received PN sequence S_P^Z to obtain the transformation matrix M. Formally, a PN sequence i in S_D^Z and S_P^Z can be represented as $s_D^Z(i)$ and $s_P^Z(i)$, respectively. The element in the ith row jth column of M is denoted as m_{ij}. Then, we have:

$$\begin{bmatrix} s_P^Z(1) \\ \vdots \\ s_P^Z(n) \end{bmatrix} = \begin{bmatrix} m_{11} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & m_{nn} \end{bmatrix} \begin{bmatrix} s_D^Z(1) \\ \vdots \\ s_D^Z(n) \end{bmatrix} \quad (1)$$

As shown in this equation, the matrix M ***dynamically*** reveals the impact of the imperfect emulation and the wireless channel conditions. To calculate each element m in M, it is necessary to quantize $s_D^Z(i)$ and $s_P^Z(i)$. In KEP, we use the complex values of the samples in $s_D^Z(i)$ and the chips in each $s_P^Z(i)$ for quantization. However, since the sampling rate of WiFi is $20MHz$ while the chips rate of ZigBee is $2MHz$, the number of samples in $s_D^Z(i)$ is 10 times as high as the number of chips in $s_P^Z(i)$. As a result, the mismatches between $s_D^Z(i)$ and $s_P^Z(i)$ make the above equation hard to solve.

To overcome this challenge, we introduce a chip extension scheme to extend the number of chips in $s_P^Z(i)$. Specifically, for a single chip $\alpha_j(i)$ in the actual received PN sequence $s_P^Z(i)$, we calculate the Euclidean Distances between $\alpha_j(i)$ and the corresponding 10 samples $(\beta_{j-4}(i), ..., \beta_{j+5}(i))$ in the desired PN sequence $s_D^Z(i)$. The distances between $\alpha_j(i)$ and these 10 samples are represented as $d_{j-4}(i), d_{j-3}(i), ..., d_j(i), ..., d_{j+5}(i)$. Then, to extend the chip $\alpha_j(i)$, KEP calculates the weight w of each distance using

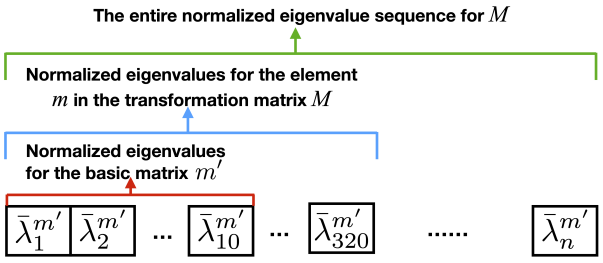


Figure 8: A normalized eigenvalue sequence generated by Bob.

$w_{d_j(i)} = \frac{d_j(i)}{\sum_{j=j-4}^{j=j+5} d_j(i)}$. At last, the corresponding extended chips $\alpha_j(e)$ can be calculated as $\alpha_j(e) = w_{d_j(i)} d_j(i) + \alpha_j(i)$. By using this approach, a single chip is extended to 10 chips.

4.2.2 ***Transformation Matrix Generation.*** Based on the extended chips, a 10×10 matrix m' that transfers 10 WiFi samples to extended chips can be represented as:

$$\begin{bmatrix} \alpha_{j-4}(i) \\ \vdots \\ \alpha_{j+5}(i) \end{bmatrix} = \begin{bmatrix} m'_{11} & \dots & m'_{1,10} \\ \vdots & \ddots & \vdots \\ m'_{10,1} & \dots & m'_{10,10} \end{bmatrix} \begin{bmatrix} \beta_{j-4}(i) \\ \vdots \\ \beta_{j+5}(i) \end{bmatrix} \quad (2)$$

In this equation, the matrix m' is defined as the **basic matrix**, which represents the estimated channel from WiFi to ZigBee. Bob can solve m' by using the Least Squares Estimation approach [3]. Since the element m in the transformation matrix M is composed by multiple m', M can be calculated by conducting chips extension for each chip and solving basic matrices.

4.3 Key Generation and Delivery

In this section, we first introduce Feature Extraction and Key Generation process. Then, we show how to deliver the key.

4.3.1 ***Feature Extraction.*** KEP utilizes the transformation matrix M for Key Generation. Specifically, for each basic matrix m', Bob first calculates its corresponding eigenvalues $\lambda_1^{m'}, ...\lambda_{10}^{m'}$. Since the element m in the transformation matrix M is composed of multiple m', the legitimate ZigBee device Bob can get a sequence of eigenvalues for each element m by calculating the eigenvalues of basic matrixes. These eigenvalues can be considered as the vibrations of the the wireless channel and the measurement of PSR.

Based on these eigenvalues, Bob will conduct normalization to get the normalized eigenvalues $\bar{\lambda}_i^{m'}$. The normalization process can be done by using: $\bar{\lambda}_i^{m'} = \frac{\lambda_i^{m'} - \lambda_{min}^{m'}}{\lambda_{max}^{m'} - \lambda_{min}^{m'}}$. Then, Bob will construct a normalized eigenvalue sequence which contains all the normalized eigenvalues of the transformation matrix M. A simplified example is shown in Figure 8. Each box contains a normalized eigenvalue and every 10 values correspond to one basic matrix m'. Since a ZigBee chip sequence contains 32 chips, every 320 normalized eigenvalues correspond to an element m in the transformation matrix M. Therefore, final sequence contains all the normalized eigenvalues corresponding to all the elements in the transformation matrix M. At this point, since Eve receives a different PN sequence, the normalized eigenvalues generated by Eve are different.

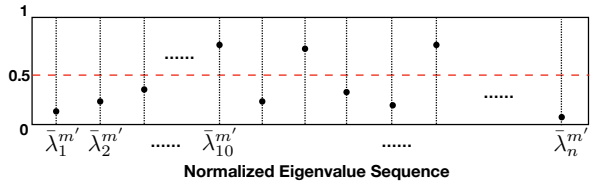


Figure 9: Each normalized eigenvalue in the sequence is compared with $q = 0.5$. If it is larger than q, the corresponding secret bit will be 1. Otherwise, the secret bit will be 0.

4.3.2 ***Key Generation.*** In KEP, Bob generates the secret key based on the normalized eigenvalues. Assume the sequence of normalized eigenvalues generated by Bob is $\bar{\lambda}_1^{m'}, ..., \bar{\lambda}_n^{m'}$. Based on this sequence, Bob will create a quantizer $Q(\bar{\lambda})$ that serves as the reference levels for each normalized eigenvalue, which can be modeled as:

$$Q(\bar{\lambda}) = \begin{cases} 1 & (\bar{\lambda} > q) \\ 0 & (\bar{\lambda} \leq q) \end{cases} \tag{3}$$

According to this quantizer, Bob will check each normalized eigenvalue $\bar{\lambda}$ to determine the corresponding secret bit. For the normalized eigenvalue that is larger than q, its corresponding secret bit will be 1. Otherwise, its corresponding secret bit will be 0. In practice, the value of q can be determined by the distribution of the normalized eigenvalue in order to make the distribution of 0 and 1 more random.

A simplified example is shown in Figure 9. q is set to 0.5 and each eigenvalue is represented with the secret bit 0 or 1 accordingly. In practice, the value of q can be determined by the distribution of the normalized eigenvalue in order to make the distribution of 0 and 1 more random. We also need to mention that due to the imperfect emulation, some of the desired PN sequence $s_D^Z(i)$ may not be recognized as the ZigBee signal. In this case, there will be no corresponding actual received PN sequence $s_P^Z(i)$ at Bob's side. As a result, Bob cannot calculate the normalized eigenvalues for these unrecognized PN sequence. In our protocol, Bob uses some predefined bit strings (e.g., 0000) to represent these unrecognized ZigBee signals.

4.3.3 ***Key Delivery.*** To deliver secret keys to Alice, the key idea is to use the packet-level ZigBee-to-WiFi CTC technique, which has been introduced in ZigFi [18] and Amphista [37], etc. Specifically, Bob conducts ZigBee-to-WiFi (Z2W) communication by varying its transmission power. For the WiFi device Alice, it can demodulate the message by measuring the amplitude of the Channel State Information (CSI). On the contrary, since Eve is located at a difference place, the CSl measurement at Eve's side is different. Moreover, even if Eve has complete knowledge of the locations of Bob and Alice, due to the random spatial and temporal variation of the wireless channel, it is still difficult for Eve to estimate the Alice's CSI.

In practice, to initialize Z2W CTC, Bob and Alice are required to coordinate with each other to determine the optimal power range, which may leak some useful information to Eve. To secure the Z2W CTC, KEP uses the actual received PN sequence to estimate optimal power range. Specifically, the ZigBee device Bob senses the Received Signal Strength (RSS) for the received PN sequences. Then, Bob will estimate the channel distortions based on the RSS value. Since Bob will send the response immediately after receiving the actual received PN sequence, the variations of channel distortions can be ignored. Therefore, Bob can predict the channel distortions with relatively low errors. Formally, the power level P_{RX}^w detected by Alice can be formulated as:

$$P_{RX}^w = P_{TX}^z + G^z - L^z - L_{FS} - L_M + G^w - L^w \tag{4}$$

where P_{TX}^z, G^z and L^z represent the transmission power, antenna gain, and transmitter losses of Bob, respectively. L_{FS} and L_M denote the path losses and miscellaneous losses while G^w and L^w represent the antenna gain and the transmitter losses of Alice. Theoretically, since Bob already knows its own transmission power P_{TX}^z, antenna gain G^z and transmitter losses L^z, it only needs to estimate L_{FS}, L_M, G^w and L^w. However, in practice, since Alice and Bob's antennas are not strictly calibrated, they cannot estimate the antenna gain and transmitter losses accurately. To overcome this challenge, in KEP, Bob leverages the detected power level P_{RX}^z from the WiFi-to-ZigBee CTC. Specifically, during the WiFi Symbol Sequence Transmission process, the power level detected by Bob can be represented as:

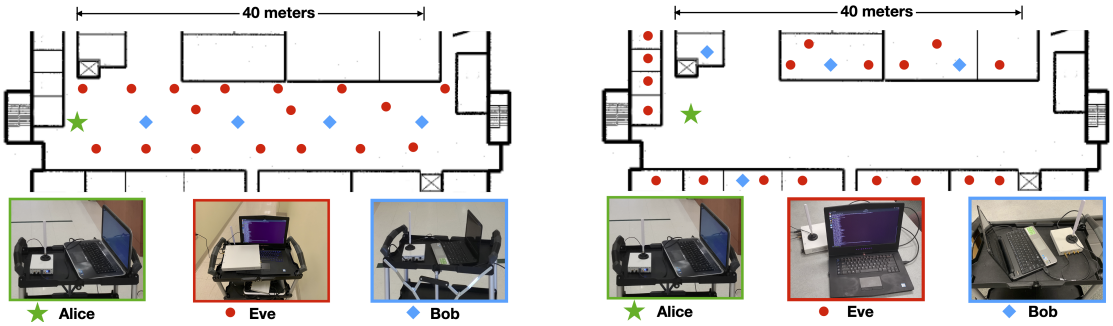
$$P_{RX}^z = P_{TX}^w + G^w - L^w - L_{FS} - L_M + G^z - L^z \tag{5}$$

where P_{TX}^w represents the transmission power of Alice. From equation 4 and 5, the power level P_{RX}^w detected by Alice can be estimated by using $P_{RX}^w = P_{TX}^z + P_{RX}^z - P_{TX}^w$. Since the transmission power of Alice P_{TX}^w is normally set to a fixed value (i.e., $0dBm$, $10dBm$ and $20dBm$), by controlling the transmission power P_{TX}^z, Bob can estimate the detected power level at the Alice's side. On the contrary, it is difficult for Eve to estimate G^w, L^w, L_{FS}, L_M, G^z, and L^z at the same time. Even if we assume that Eve knows the models of the antennas (i.e., Eve knows G^w and G^z), the transmitter losses, path losses, and miscellaneous losses still vary from device to device. Therefore, Eve cannot get the secret keys by eavesdropping the communication.

4.4 Communication Compensation

In KEP, we introduce a communication compensation scheme during the key establishment process to simultaneously overcome the unreliability introduced by the imperfect emulation and dynamic wireless environments. Specifically, since KEP uses predefined bit strings to represent the unrecognized PN sequences, Alice will check if the received secret key contains these bit strings and discard the corresponding WiFi symbol combination in $s_D^Z(i)$. As mentioned in the section 4.1, the WiFi device can use multiple different combinations to emulate a single ZigBee PN sequence. Alice can always select the combination with the minimal Euclid Distance between the WiFi QAM point and the desired ZigBee signal for signal emulation.

We also need to mention that it is possible that all the corresponding WiFi symbol combinations for some specific ZigBee PN sequences cannot be recognized as ZigBee signals. In this case, Alice will use the WiFi symbol combinations that have been recognized as the ZigBee signal to represent these PN sequences. For example, assume the ZigBee PN sequences PN_1 and PN_2 can be emulated by the WiFi symbol combinations $S_1S_2S_3S_4$ and $S_5S_6S_7S_8$, respectively. After receiving the secret key, Alice finds out that the emulated



(a) The deployment in LoS scenario.

(b) The deployment in NLoS scenario.

Figure 10: Evaluation setup in the smart building scenario.

signal PN_1 cannot be recognized while PN_2 can be detected by Bob. In this case, Alice will use n consecutive combinations of $S_5S_6S_7S_8$ to represent PN_1 for reliable communication. For the ZigBee device Bob, once it receives n consecutive PN sequences PN_1, it will consider these sequences as PN_1.

5 SECURITY ANALYSIS

In this section, we discuss how KEP defends against three attacks: Eavesdropping, Spoofing and Predictable Channel attacks.

• **Eavesdropping.** Eve is located at a different place and is passively eavesdropping the communication between Alice and Bob. As mentioned in section 2.1, for the communication from Alice to Bob, the PSR distributions between Eve and Bob are different. In other words, Eve cannot get any useful information for key generation. For the communication from Bob to Alice, since Eve has independent channel variations, it is hard for Eve to predict the change of CSI values. Moreover, the communication compensation is based on Bob's demodulation results, which is useless to Eve. As a result, Eve cannot get any useful information and will also suffer high packet loss ratio.

• **Spoofing Attack.** For the spoofing attack, we assume Eve may somehow manage to get the secret key. Then, Eve can actively impersonate Alice (WiFi) or Bob (ZigBee) without revealing its presence. Our approach can effectively detect Eve in this scenario. For example, if Eve transmits the fake messages S_D^Z to Bob, Bob can compare the desired PN sequence and the actual received PN sequence S_P^Z to detect the potential attack. Specifically, the ZigBee device demodulates the emulated signal based on the phase information. The differences between S_D^Z and S_P^Z reveal the phase offset between the emulated signal and the desired signal. The major contributions to the phase offset are imperfect emulation, channel variations, hardware defects (i.e., carrier frequency offset f_c, sampling frequency offset f_s, detection delay t_d and phase lock loop error θ_P).

Formally, the phase of the desired signal can be represented as θ_d while the phase offset introduced by channel variations can be represented as θ_c. Therefore, the phase of the actual received signal θ_z at time t can be calculated as: $\theta_z = \theta_d - \theta_c - (2\pi f_c t + 2\pi f_s t_d + \theta_P)$. Since θ_z and θ_d are known by Bob, Bob can analyze the channel variations and hardware defects by simply using $\theta_z - \theta_d$. However, when Eve transmits the emulated PN sequence S_e^W to Bob, some of the received PN sequences at Bob's side will be significantly different from the prediction. Then, Bob may consider that a potential attacker is trying to conduct the spoofing attack. Similarly, when Eve impersonates Bob, it does not know the accurate RSS or CSI information measured by Alice. Therefore, Eve can only make guesses to vary its transmission power. As a result, Alice will experience high bit error rate, which implies that the spoofing attack may be occurring. Therefore, by setting the proper Bit Error Rate Threshold, Alice may detect the spoofing attack. Even if we assume that Eve luckily transmits some messages to Alice with a low bit error rate, Alice can still detect the spoofing attack by checking the PSR of the received messages. The abnormal PSRs may imply a potential spoofing attack.

• **Predictable Channel Attack.** Eve may also want to create desired and predictable changes in the channel by making planned movements between Alice and Bob. This attack mainly focuses on the prediction of RSS or CSI value, which makes RSS-based or CSI-based key generation methods vulnerable [20]. However, since KEP utilizes PSR for key extraction, it will be harder for Eve to predict the PSR changes. Specifically, instead of converting CSI or RSS into bits, KEP converts the actual received signals (ZigBee chips) into bits. Since the ZigBee device updates its chip every $1\mu s$, it is much faster than the update rate of CSI (4 μs for WiFi) and RSS (i.e., the update rate is at millisecond level). Therefore, KEP is more sensitive than the coarse CSI or RSS measurement, which makes it hard for the attacker to conduct predictable channel attacks.

6 IMPLEMENTATION AND EVALUATION

We extensively evaluate the performance of KEP under various scenarios and settings. In this section, we first introduce the experiment implementation. Then, we show the experiment results and the security insights of KEP.

6.1 Implementation

We implement the legitimate WiFi device (Alice) using a WiFi compliant USRP B210 [6]. Alice utilizes the physical layer emulation technique to communication with the ZigBee device (Bob) [22]. To extensively evaluate KEP and get physical layer raw data, we also implement the Bob part of KEP on USRP B210 according to the 802.15.4 standard. The Attacker (Eve) is implemented using a high performance USRP X300 with a UBX160 USRP Daughter Board

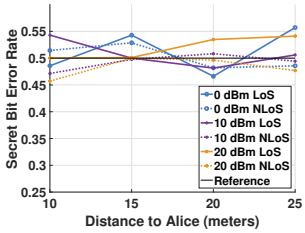


Figure 11: Secret Bit Error Rate

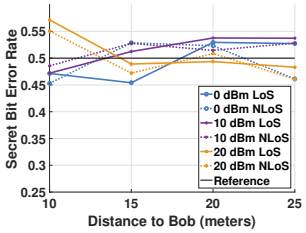


Figure 12: Secret Bit Error Rate

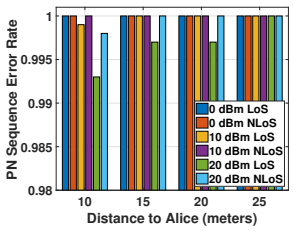


Figure 13: Eavesdropping PN Sequence Error Rate

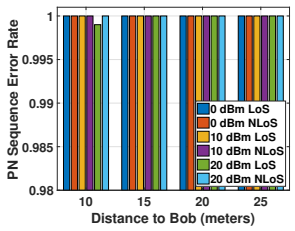


Figure 14: Eavesdropping PN Sequence Error Rate

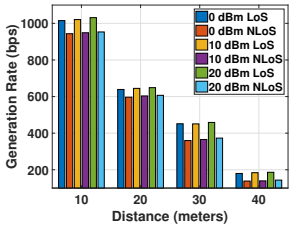


Figure 15: Secret Bit Generation Rate

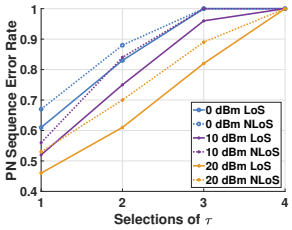


Figure 16: Selections of τ.

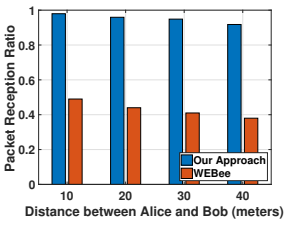


Figure 17: Packet Reception Ratio (LoS)

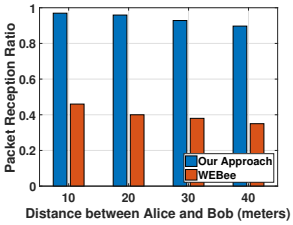


Figure 18: Packet Reception Ratio (NLoS)

[5]. The transmission power of Alice varies between $0dBm$, $10dBm$ and $20dBm$ and the τ is set to 4. To evaluate the communication reliability, we choose the most related physical-layer emulation approach - WEBee [22] as our baseline.

We evaluate KEP in the smart building scenario with Line-of-Sight (LoS) and Non-Line-of-Sight (NLoS) settings. In Figure 10a, Alice, Bob and Eve are in the LoS scenario while Bob and Eve are placed in different rooms (NLoS scenario) in Figure 10b. The distance between Alice and Bob varies from $10m$ to $40m$. To effectively attack Alice and Bob and remain stealthy, the distances of Eve to Alice and Eve to Bob vary from $10m$ to $25m$. Each experiment is repeated 1000 times and we show the average value of the experiment results.

6.2 Evaluation Metrics

In this work, we define four metrics to evaluate KEP.

• **Secret Bit Generation Rate.** In KEP, Secret Bit Generation Rate (sBGR) is defined as the number of secret bits generated over the entire time of the key establishment process, including WiFi Symbol Sequence Transmission, Transformation Matrix Generation, Key Generation, and Key Delivery.

• **Eavesdropping PN Sequence Error Rate.** The adversary may eavesdrop the transmissions from Alice to Bob. To evaluate the effectiveness of KEP, we define the Eavesdropping PN Sequence Error Rate (EPR) as the number of the wrong PN sequences (PN_e) received by the adversary (Eve) over the total number of actual received PN sequences (PN_b) at Bob's side, which can be calculated as: $EPR = \frac{PN_e}{PN_b}$.

• **Secret Bit Error Rate.** The Secret Bit Error Rate (sBER) is defined as the number of error bits received by Eve over the number of bits transmitted from Bob.

• **Spoofing Attack Detection Rate.** We define the Spoofing Attack Detection Rate as the number of detected spoofing attack tests over the number of attack tests.

6.3 Evaluation Results

6.3.1 ***Secret Bit Error Rate.*** Figure 11 and Figure 12 show the Secret Bit Error Rate (sBER) for Eve in both LoS and NLoS scenarios. The transmission power of Alice varies from $0dBm$ to $20dBm$. The sBER for all these experiments are close to 50% regardless of the changing of transmission powers and the communication distances. This is because Eve's PSRs are different from Bob's PSRs. Since the imperfect emulation is introduced by the limitations of the 802.11 physical layer and the variations of wireless environments, the increase of transmission power or decrease of communication distances cannot eliminate the imperfect emulation. As a result, the performance of eavesdropping is similar to a random guess.

6.3.2 ***Eavesdropping PN Sequence Error Rate.*** Figure 13 and Figure 14 show the experiment results of Eavesdropping PN Sequence Error Rate (EPR). The transmission power of Alice also changes from $0dBm$ to $20dBm$. As shown in these two figures, the Eavesdropping PN Sequence Error Rates are almost 1 for both cases, which means the actual received PN sequences for Eve are totally different from Bob. These two experiments explain why the sBER for Eve are around 50%. Since the actual received PN sequences are different from Bob, Eve cannot get the correct normalized eigenvalues for key generation. We can also observe that as transmission power of Alice increases and distance decreases, the EPRs for both cases are slightly reduced. However, even for the worst case, the EPR is still as high as 0.993, which is still not enough for Eve to get the correct secret bits.

6.3.3 ***Secret Bit Generation Rate.*** Figure 15 shows the secret bit generation rate (sBGR) under different communication distances and transmission powers. Specifically, when the distance between Alice and Bob is $10m$, the secret bit generation rates are higher than $1000bits/sec$ and $900bits/sec$ for LoS and NLoS scenario, respectively. Therefore, for both cases, it only takes Bob no more than 0.3 seconds to generate a 256-bit key. When the distance reaches

Test	P-value
Monobit Frequency	0.7237
Runs Test	0.8509
Approx. Entropy	0.9301
Random Excursions	0.5868

Table 1: The randomness tests on secret bits

$40m$, the secret bit generation rates for LoS and NLoS are around $200bits/sec$ and $130bits/sec$, respectively. This is because the communication rate between WiFi and ZigBee reduces as distances increase. However, even in this scenario, it still only takes around 2 seconds to finish the key establishment. Compared with traditional key agreement approach that requires tens of seconds [23], our approach is much faster and efficient.

6.3.4 ***The selections of*** τ. Figure 16 shows the EPR with different selections of τ. In this experiment, Eve is placed $20m$ from Alice and the transmission power of Alice is set to $10dBm$. We can observe that the EPR increases rapidly with the increase of τ. This is because a bigger τ gives Alice more choices to determine the WiFi symbols for ZigBee PN sequence emulation. In addition, a bigger τ also increases the number of WiFi symbols in the WiFi Symbol sequence S_e^W, which increases communication uncertainty at Eve's side. We also need to mention that even if τ is set to 1, EPR is still around 50% in the worst case (20 dBm NLoS). This is because Eve is located at a different position. As mentioned in section 2.2, *the PSRs are similar if and only if the receivers are very close to each other.* Therefore, even in this case, Eve is still suffering high uncertainty.

6.3.5 ***Communication Reliability.*** Figure 17 and Figure 18 show the comparisons of Packet Reception Ratio (PRR) between KEP and WEBee. In these experiments, the packet size is set to 48 Bytes. As we can see from these figures, the packet reception ratio of KEP shows great advantages over WEBee. This is because the key establishment process of KEP offers an opportunity for the WiFi to better understand the channel conditions and the imperfect emulation. Moreover, for the unrecognized emulated signal, the WiFi device can leverage multiple compensated symbols to conduct reliable WiFi-to-ZigBee communication. On the contrary, WEBee does not have the feedback process. As a result, the PRR for KEP is around **2** times as high as that of WEBee.

6.3.6 ***Randomness of Generated Secret Bits.*** We study the randomness of the generated bits in Table 1. To conduce this experiment, we ran a few tests using the NIST test suite [9]. In each test, if a p-value is larger than 0.01, the generated secret bits should be considered as random bits. As we can see from the testing results, the secret bits generated by KEP show high randomness. In particular, the approximate entropy is very close to 1, which means that the generated sequence follows the randomness requirements in NIST. This is because KEP not only takes the imperfect emulation into consideration but also considers the randomness of the wireless channel. Therefore, we believe that the KEP is sufficient to secure the asymmetric CTC between WiFi and ZigBee.

6.3.7 ***Spoofing Attack Detection Evaluation.*** We study the performance of KEP against the spoofing attack. In these experiments, Eve is placed $20m$ from Alice. It can either impersonate Alice to perform key establishment with Bob or impersonate Bob to deliver wrong keys to Alice. The transmission power of Alice is set to $10dBm$.

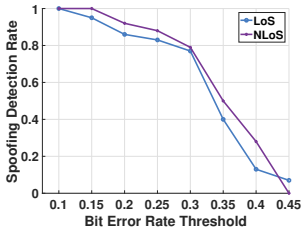


Figure 19: Attack Detection Rate

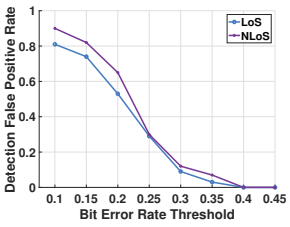


Figure 20: False Positive Rate

In Figure 19. we show the Spoofing Attack Detection Rate with different Bit Error Rate Thresholds. In this experiment, Eve impersonates Bob and tries to transmit wrong secret keys to Alice. As shown in this figure, the Spoofing Detection Rate reduces as the Bit Error Rate Threshold increases. When the Bit Error Rate Threshold reaches 0.45, the Spoofing Detection Rate is lower than 0.1 (10%). This is because a high Bit Error Rate Threshold allows Eve to make more 'mistakes' without being identified as the spoofing attack. However, in this case, due to the high bit error rate, since the Alice and Bob also cannot perform reliable cross-technology communication, they can restart the key establishment process and communication compensation process to eliminate the harmful impacts of Eve.

In Figure 20, we also study the corresponding Detection False Positive Rates with different Bit Error Rate Thresholds. As we can observe in this figure, although a lower Bit Error Rate Threshold (0.1) will give Alice a higher Spoofing Detection Rate, the Detection False Positive Rate is also higher than 0.8. This is because Alice misclassifies the communication from legitimate ZigBee to WiFi device as potential spoofing attack. As the Bit Error Rate Threshold increases, the Detection False Positive Rate reduces rapidly. When the Bit Error Rate Threshold reaches 0.3, the Detection False Positive Rate drops to around 0.08. However, even in this case, the Spoofing Attack Detection Rate is still as high as 0.8 in Figure 19. This experiment proves our security analysis in section 5, by setting a proper Bit Error Rate Threshold, Alice can detect the spoofing attack with a low Detection False Positive Rate.

7 RELATED WORK

The related work can be categorized as two parts: **I.** Cross-technology Communication and **II.** Secret Key Generation.

I. Cross-technology Communication: Researchers have proposed many innovative techniques that enable CTC among heterogeneous IoT devices (e.g., WiFi, ZigBee, and Bluetooth). Pioneering works mainly use packet-level modulation to achieve CTC, such as FreeBee [21], EMF [12], B^2W^2 [14], and C-morse [34], etc. For example, FreeBee [21] enables communication between WiFi and ZigBee by modulating the intervals of WiFi beacons. B^2W^2 [14] uses the WiFi CSI to demodulate the BLE data while C-morse [34] uses WiFi to transmit long and short packets to construct Morse codes. However, since the duration of a wireless packet is in the range of milliseconds, the throughput and communication range of packet-level CTC are limited. To improve the throughput, the most recent and

widely applied work – WEBee [22] first proposes the physical layer emulation technique to directly conduct communication from WiFi to ZigBee. However, due to the limitations of 802.11 physical layer, the emulated signal cannot perfectly match the desired ZigBee signal, which sacrifices the communication reliability.

II. Secret Key Generation: Channel characteristics have been studied by many researchers for key generation [10, 24, 26, 28, 30]. For example, [10] uses a rotating directional antenna to generate secret bits between two ZigBee nodes. [24] and [28] use multiple characteristics such as channel impulse response and amplitude measurements for key agreement. The spatial correlation is also studied for key generation [16]. Hu-Fu [30] uses inductive coupling of two adjacent RFID tags and signal randomization to provide physical layer security.

Different from the above approaches, KEP is the first study on the security problem with the coexistence of heterogeneous IoT devices (i.e., WiFi and ZigBee). In our work, we discover PSR and utilize it for CTC key establishment. Moreover, instead of sacrificing the performance of CTC, KEP almost doubled the communication reliability at the same time.

8 CONCLUSION

This paper presents KEP, an advanced key establishment protocol that mainly focuses on heterogeneous IoT devices to support secure WiFi and ZigBee communication. Compared with traditional key establishment solutions, KEP has four unique advantages: **i)** it is the first work that focuses on the security issues in CTC; **ii)** it in-depth explores the WiFi-to-ZigBee CTC and discovers an unique feature called PSR for secret key establishment; **iii)** it can generate a strong secret key and effectively defend against multiple attacks; **iv)** it can double the communication reliability at the same time. We conduct extensive experiments in real-world settings to evaluate KEP and the results prove the effectiveness of our design. We believe KEP will not only allow more secure and reliable communication among heterogeneous IoT devices but also open a new direction for future secure CTC techniques.

9 ACKNOWLEDGEMENT

This work is partially supported by NSF grants CNS-2305246, CNS-2316605, and CNS-2127881.

REFERENCES

[1] 2020. http://www.gartner.com/newsroom/id/3598917.
[2] 2023. http://edadocs.software.keysight.com/display/sv201001/About+ZigBee+Baseband+Verification+Library.
[3] 2023. https://en.wikipedia.org/wiki/Least_squares.
[4] 2023. https://en.wikipedia.org/wiki/Reciprocity_(electromagnetism).
[5] 2023. https://www.ettus.com/all-products/x300-kit/.
[6] 2023. https://www.ettus.com/product/.
[7] 2023. https://www.healthline.com/health/extreme-temperature-safety.
[8] 2023. https://www.livestrong.com/article/552385-dangerous-temperatures-for-the-elderly/.
[9] 2023. https://www.nist.gov/publications/statistical-test-suite-random\-and-pseudorandom-number-generators-cryptographic.
[10] Tomoyuki Aono, Keisuke Higuchi, Takashi Ohira, Bokuji Komiyama, and Hideichi Sasaoka. 2005. Wireless secret key generation exploiting reactance-domain scalar response of multipath fading channels. IEEE Transactions on Antennas and Propagation (2005).
[11] Yongrui Chen, Zhijun Li, and Tian He. 2018. TwinBee: Reliable Physical-Layer Cross-Technology Communication with Symbol-Level Coding. In IEEE INFOCOM.
[12] Z. Chi, Z. Huang, Y. Yao, T. Xie, H. Sun, and T. Zhu. 2017. EMF: Embedding multiple flows of information in existing traffic for concurrent communication among heterogeneous IoT devices. In IEEE INFOCOM.
[13] Zicheng Chi, Yan Li, Xin Liu, Yao Yao, Yanchao Zhang, and Ting Zhu. 2019. Parallel inclusive communication for connecting heterogeneous IoT devices at the edge. In Proceedings of the 17th conference on embedded networked sensor systems.
[14] Zicheng Chi, Yan Li, Hongyu Sun, Yao Yao, Zheng Lu, and Ting Zhu. 2016. B2W2: N-Way Concurrent Communication for IoT Devices. In SenSys.
[15] Deepa Devarajan, Gursharan Singh, and Pooja Gupta. 2019. Security Analysis of Encrypted Key Exchange with Diffie Hellman in Wireless Sensor Networks. Our Heritage (2019).
[16] M. Gapeyenko, A. Samuylov, M. Gerasimenko, D. Moltchanov, S. Singh, M. Akdeniz, E. Aryafar, S. Andreev, N. Himayat, and Y. Koucheryavy. 2019. Spatially-Consistent Human Body Blockage Modeling: A State Generation Procedure. IEEE TMC (2019).
[17] Balamurugan Gopalakrishnan and Marcharla Anjaneyulu Bhagyaveni. 2016. Random codekey selection using codebook without pre-shared keys for anti-jamming in WBAN. Computers & Electrical Engineering (2016).
[18] Xiuzhen Guo, Yuan He, Xiaolong Zheng, Liangcheng Yu, and Omprakash Gnawali. 2018. Zigfi: Harnessing channel state information for cross-technology communication. In IEEE INFOCOM.
[19] Xiuzhen Guo, Xiaolong Zheng, and Yuan He. 2017. Wizig: Cross-technology energy communication over a noisy channel. In IEEE INFOCOM.
[20] Suman Jana, Sriram Nandha Premnath, Mike Clark, Sneha K Kasera, Neal Patwari, and Srikanth V Krishnamurthy. 2009. On the effectiveness of secret key extraction from wireless signal strength in real environments. In MobiCom.
[21] Song Min Kim and Tian He. [n. d.]. FreeBee: Cross-technology Communication via Free Side-channel. In MobiCom '15.
[22] Zhijun Li and Tian He. [n. d.]. WEBee: Physical-Layer Cross-Technology Communication via Emulation. In MobiCom '17.
[23] Suhas Mathur, Robert Miller, Alexander Varshavsky, Wade Trappe, and Narayan Mandayam. 2011. Proximate: proximity-based secure pairing using ambient wireless signals. In Proceedings of the 9th international conference on Mobile systems, applications, and services.
[24] Suhas Mathur, Wade Trappe, Narayan Mandayam, Chunxuan Ye, and Alex Reznik. 2008. Radio-telepathy: extracting a secret key from an unauthenticated wireless channel. In MobiCom.
[25] Sriram N. Nandha Premnath, Suman Jana, Jessica Croft, Prarthana L. Lakshmane Gowda, Mike Clark, Sneha K. Kasera, Neal Patwari, and Srikanth V. Krishnamurthy. 2013. Secret Key Extraction from Wireless Signal Strength in Real Environments. IEEE Transactions on Mobile Computing (2013).
[26] Akbar Sayeed and Adrian Perrig. 2008. Secure wireless communications: Secret keys through multipath. In 2008 IEEE International Conference on Acoustics, Speech and Signal Processing. 3013–3016.
[27] Wenlong Shen, Bo Yin, Xianghui Cao, Lin X Cai, and Yu Cheng. 2016. Secure device-to-device communications over WiFi direct. IEEE Network (2016).
[28] Hailun Tan, Diet Ostry, and Sanjay Jha. 2018. Exploiting multiple side channels for secret key agreement in Wireless Networks. In ACM ICDCN.
[29] Ángeles Vázquez-Castro. 2022. Asymptotically guaranteed anti-jamming spread spectrum random access without pre-shared secret. IEEE Transactions on Information Forensics and Security (2022).
[30] Ge Wang, Haofan Cai, Chen Qian, Jinsong Han, Xin Li, Han Ding, and Jizhong Zhao. [n. d.]. Towards Replay-resilient RFID Authentication. In MobiCom '18.
[31] Tao Wang, Yao Liu, and Athanasios V. Vasilakos. 2015. Survey on Channel Reciprocity Based Key Establishment Techniques for Wireless Systems. Wirel. Netw., 2015 (2015).
[32] Wei Wang, Tiantian Xie, Xin Liu, and Ting Zhu. 2018. Ect: Exploiting cross-technology concurrent transmission for reducing packet delivery delay in iot networks. In IEEE INFOCOM 2018-IEEE Conference on Computer Communications.
[33] Wei Xi, Chen Qian, Jinsong Han, Kun Zhao, Sheng Zhong, Xiang-Yang Li, and Jizhong Zhao. 2016. Instant and Robust Authentication and Key Agreement Among Mobile Devices. In CCS '16.
[34] Z. Yin, W. Jiang, S. M. Kim, and T. He. 2017. C-Morse: Cross-technology communication with transparent Morse coding. In IEEE INFOCOM.
[35] Zhimeng Yin, Zhijun Li, Song Min Kim, and Tian He. 2018. Explicit Channel Coordination via Cross-technology Communication. In Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '18).
[36] Ruogu Zhou, Yongping Xiong, Guoliang Xing, Limin Sun, and Jian Ma. 2010. ZiFi: Wireless LAN Discovery via ZigBee Interference Signatures. In MobiCom.
[37] Chi Y. Li Z. Huang H. Sun Zicheng and T. Zhu. 2019. Simultaneous Bi-directional Communications and Data Forwarding using a Single ZigBee Data Stream. In IEEE INFOCOM.