UWBKey: Using Contrastive Learning for Efficient Secure Key Generation in UWB

HAIGE CHEN, Georgia Institute of Technology, USA
ASHUTOSH DHEKNE, Georgia Institute of Technology, USA

Ultra-wideband radios are being used for a multitude of localization and ranging applications, from object finding using Apple AirTags, to industrial precise asset localization, to keyless car doors. UWB's ubiquitous presence is fueled, in part, by its effectiveness in separating the wireless signal's first path from multipath and its co-existence with Wi-Fi and other wireless technologies. Yet, today UWB-based devices are still required to be "paired" with a controller device which can gain exclusive access to communicate with that device. This pairing requirement restricts truly ubiquitous and pervasive use-cases for UWB. One of the primary reasons for the pairing requirement is to enable secure UWB communication, including the communication about the ranging parameters and timestamps which then enables secure UWB ranging. Today, this pairing is performed using an out-of-band communication mechanism based on Bluetooth, or Wi-Fi. In this work, we show that an out of band communication is not necessary for establishing secure keys for UWB communication. In fact, the channel impulse response obtained by every UWB device when receiving a packet can itself be leveraged to generate a symmetric secure key. We call our system, *UWBKey*, and have fully implemented it on a commercial off-the-shelf UWB chip (Qorvo DWM1000). We show that the channel impulse response (CIR) is reciprocal for a pair of communicating devices, and that we can derive a key from the CIR that is close to random. In contrast, an eavesdropper even when nearby cannot obtain the same CIR and therefore cannot infer the key.

This paper develops an important component of the whole UWB ecosystem within the constraints of today's UWB hardware (in terms of available bandwidth), practical environments (in terms of representative public places we could access), and practical device density (with assumptions about how near a malicious device could get). By no means do we believe we have solved the secure communication problem fully, but we are convinced that our open-data, open-hardware, open-software approach in this paper will enable a new thought process among researchers enabling UWB to stand on its own as a secure radio communication and ranging technology, without depending on out-of-band solutions.

ACM Reference Format:
Haige Chen and Ashutosh Dhekne. 2025. UWBKey: Using Contrastive Learning for Efficient Secure Key Generation in UWB. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 9, 4, Article 164 (December 2025), 31 pages. https://doi.org/10.1145/3770630

1 INTRODUCTION

Ultra-wideband (UWB) is an emerging wireless technology that has demonstrated highly accurate and robust indoor localization performance. With a bandwidth of 500 MHz and higher, it can resolve multipath separated by nanosecond delays, enabling decimeter-level ranging precision. With its low power consumption and minimal interference with other wireless technologies, UWB has been quickly adopted in several consumer mobile phone products, automotive products, and industrial solutions for a wide range of applications. For example, many leading mobile manufacturers, such as Apple [33], Google [27], Samsung [48], Motorola [50], and Xiaomi [51], have integrated UWB in the newest mobile phones, smart watches, and other IoT products to provide precise localization

Authors' addresses: Haige Chen, Georgia Institute of Technology, Atlanta, USA, hchen425@gatech.edu; Ashutosh Dhekne, Georgia Institute of Technology, Atlanta, USA, dhekne@gatech.edu.

ACM 2474-9567/2025/12-ART164
https://doi.org/10.1145/3770630

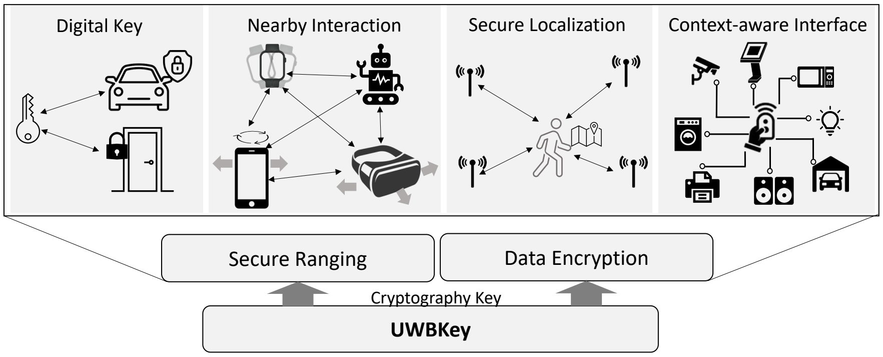


Fig. 1. *UWBKey* is a robust, on-the-fly, in-band cryptographic key generation solution for COTS UWB devices that enables seamless secure ranging and data encryption. It allows UWB to be a standalone and secure solution for digital key, nearby interaction, localization and context-aware interfacing applications.

of tagged personal belongings. Apple has envisioned the future usage of UWB for "Nearby Interactions" [32] by enabling multi-user interactive sessions using location information, which could enhance interactions with public infrastructure such as kiosks, AR experiences, provide novel gaming interfaces, and empower new applications, including those that have not been conceived yet. In the automotive industry, UWB is being integrated into the digital key solution [17] for a seamless and secure keyless entry experience. In industrial and enterprise settings, UWB is also finding applications in automation and asset tracking.

The growing adoption of UWB in these crucial applications also increases its vulnerability to potential security threats, making localization and communication security essential for robust and reliable systems. Malicious attackers may attempt to gain unauthorized access to restricted areas or disrupt services for legitimate users by spoofing UWB localization and deceive infrastructure into behaving in unintended ways. In privacy-sensitive applications such as keyless entry and contactless payments, attackers may seek to intercept data packets to steal sensitive information. Ensuring robust security measures in UWB localization and communication is imperative to safeguard against these threats. The need for security is well recognized, resulting in significant efforts dedicated to ensuring localization and communication security in UWB. The IEEE 802.15.4z standard [16] safeguards the ranging security with a cryptographically generated pseudo-random Scrambled Timestamp Sequence (STS) in each packet. Since UWB ranging relies on finding the precise time of the first path from the cross-correlation of the received signal with the expected preamble sequence, the STS field that contains a cryptographically-secure pseudo-random sequence only known to the legitimate parties guarantees that an adversary cannot infuse a malicious signal earlier in time and reduce the measured distance [38]. The standard also recommends using the 128-bit Advanced Encryption Standard (AES) for encrypting the data packet for communication security aimed at preventing application level attacks. While cryptography is the solution for both ranging and communication security, one question remains: **since the legitimate pair of transceivers needs symmetric cryptographic keys (or the initial seed to a pseudo-random number generator) for generating the STS and encrypting the message, how should the two UWB devices agree on the same key in the first place?**

There are several ways to establish a secure key between two UWB devices: (1) through a secure out-of-band (OOB) side-channel, (2) through a key exchange process in the open channel (such as used by Wi-Fi), or (3) by generating the key from a common physical source. Option (1) uses a previously established OOB secure side-channel to communicate the key—this is the option frequently used today for UWB pairing. For example, industrial UWB Digital Key products today first establish a secure BLE connection to share the initial key to be used for UWB ranging [26]. However, this method requires a different radio hardware, inducing additional cost and footprint, and yet the security remains largely dependent on the security of the OOB channel. In addition, in Nearby Interactions and indoor navigation applications, each user device might need to communicate with multiple UWB nodes in quick succession, in which cases the links can be ad-hoc and ephemeral due to movements. Requiring the user to first pair with every node that it comes in contact with can be impractical and inefficient. For option (2), a widely used key exchange protocol is the Diffie-Hellman key exchange [21], in which two parties, *Alice* and *Bob*, each use a pseudo-random number generator to produce a local key, and then engage in an exchange process in the open channel such that the final key is a combination of the two local keys, but for a third party (Eve) who is eavesdropping in the said exchange, it is computationally impossible to obtain the same key. However, Diffie-Hellman is shown to be vulnerable to adversary with high computation powers [1], and is known to be susceptible to quantum-computing attacks as it is based on discrete logarithm problem [54]. In option (3), provided that there is a common signal source that is only observable to Alice and Bob but not to Eve, Alice and Bob can each independently generate identical secret keys from that signal source. If such a signal source is readily available, this simpler key agreement scheme would be more suitable especially for applications that require ad-hoc, on-the-fly operation and need to run on resource-constrained hardware.

In this work, we explore a key generation method that leverages the reciprocity of the wireless channel, enabling UWB secure localization and communication without the need of an OOB channel. The wireless channel between Alice and Bob (devices that wish to communicate with each other) at any particular instance is a manifestation of the propagation of the RF signal in the physical world. Since the propagation characteristics of the wireless channel are identical in both directions between Alice and Bob (just like a mirror), they should observe symmetric channel impulse response (CIR). On the other hand, Eve who is located at a different location from Alice and Bob would observe a different wireless channel. This paper investigates how the rich UWB channel impulse response can be used as a source for generating UWB secret keys on-the-fly. To validate our intuitions, we conducted a simple experiment with two devices, one stationary and the other in motion. The correlation coefficient and root-mean-square-error (RMSE) between the CIR measurement obtained at the two devices show promise (see Figure 3). The reciprocally measured CIRs show high levels of correlation, while CIRs measured after longer intervals decorrelate from each other, demonstrating both spatial and temporal variations.

Of course, designing an effective key generation and quantization function, which maps the CIR—a continuous-valued signal—to a binary-valued sequence, is a non-trivial task. Although the UWB CIR is a suitable random signal source with high reciprocity and spatial-temporal randomness, **there are several practical challenges** in deriving high quality binary keys from it. **First**, CIR obtained by a practical UWB transceiver is subject to measurement errors such as noise, sampling error, quantization error, etc. In addition, there is always an offset of a few milli-seconds between the CIR measured by Alice and Bob since they need to exchange packets one after the other. Therefore, although theoretically wireless channel is reciprocal, the measurement error and channel variation over the messaging delay can cause the measured CIR to differ. **Second**, although wireless channel de-correlates spatially, it is not guaranteed that the generated keys are random with high entropy. The UWB CIR especially has predictable structures (i.e. path arriving earlier likely has higher amplitude), and designing a key generation and quantization scheme that produces high entropy is non-trivial. **Third**, key generation from wireless channel becomes insecure when the channel becomes deterministic or contains deterministic components. This could result from a channel sparse in multipath (such as outdoors in an open area) or due to the presence of deliberately placed RF reflectors by the adversary. Ensuring the randomness of the generated keys in a deterministic channel environment is quite challenging.

Given these challenges, we hypothesize that the raw CIR may not be the most effective representation for key extraction, particularly when the aim is to achieve strong reciprocity, high entropy, and resilience against channel injection attacks. This raises a crucial question: Can we transform the raw CIR into a more suitable representation space that better facilitates secure key generation?

In other words, using the CIR as is, for deriving keys, will result in extremely weak keys with low entropy. Instead, we need a mapping function that can take as input the obtained CIR and produce a robust key. We draw inspiration from recent advances in representation learning and leverage contrastive learning to design a key mapping function that transforms the measured CIR into a latent space representation before quantizing into bits, where the latent features exhibit better reciprocity and randomness than the original CIR. Contrastive learning [35] is a machine learning technique where a model learns representations by contrasting similar (positive) and dissimilar (negative) data samples. The key idea is to pull similar instances closer in the representation space while pushing dissimilar instances farther apart. To illustrate, consider a simple image classification task—say, distinguishing dogs from cats. How do we compare two samples to tell whether they belong to the same class or different classes? Directly comparing raw pixel values would be ineffective: similar pixel intensities do not guarantee semantic similarity (e.g. a dog and a cat may share the same color but still should belong to different class labels), and dissimilar pixels also do not necessarily indicate different classes (e.g. two dogs with varying fur colors or backgrounds still belong in the same class label). Instead, we can train a model to extract latent features that truly distinguish between classes, by making sure that in the latent representation space, the positive samples maintain small distances while the negative samples are separated by large margins. This learned representation becomes robust to superficial variations while remaining sensitive to semantically meaningful differences. Returning to the key generation task, by employing contrastive learning with properly chosen distance metric, sampling scheme and loss function, we aim to improve the reciprocity between latent representations used to generate keys despite errors and noise in the raw CIR measurements, while pushing the latent representation obtained by Eve to be dissimilar. Following this train of thought, we further propose combining entropy-based regularization with data augmentation on top of contrastive learning to ensure that the latent representation also satisfies high entropy and resilience against channel injection.

Based on this idea, we propose *UWBKey*, a robust, on-the-fly, in-band cryptographic key generation solution for COTS UWB devices that enables seamless secure ranging and data encryption. Our main contributions are summarized as follows:

- By leveraging contrastive learning, entropy-based regularization and data augmentation techniques, *UWBKey* learns a key mapping function that transforms the measured CIR into a latent space representation, which produces keys with enhanced reciprocity, randomness, and robustness.
- In order to make the method pertinent to existing hardware and future envisioned use-cases, we employ quantization-aware training for model compression, which allows *UWBKey* to be deployed on resource-constrained, latency-sensitive embedded devices.
- *UWBKey* is evaluated on real-world test beds in various environments and conditions, and compared against multiple baselines. *UWBKey* shows superior performance with a memory footprint of no more than 51 kBytes, latency of no more than 50 ms when run on a COTS microcontroller, a median bit error rate of $\sim 6.3\%$, and successfully passes the NIST randomness testing [6].

What kind of use-cases will such a security primitive enable? This paper explores a fundamental security primitive without being constrained by specific applications. Yet, it is natural to wonder about the potential use-cases that *UWBKey* could enable. Imagine walking to a kiosk at an airport and being able to control it using your mobile phone, including scrolling and searching by using swipe gestures and the keyboard on your phone. Or imagine your phone instantly becoming the remote controller for any TV in the vicinity, or being able to control the smart devices in a conference room on your phone simply because you are inside that room. Establishing ad hoc secure communication between pair of devices in vicinity can help bring context to many IoT interactions, enabling a world of free-flowing and seamless interactions.

The rest of the paper is structured as follows: Section 2 describes the threat model and presents the system design of our proposed solution; Section 3 discusses the implementation details of *UWBKey*; Section 4 and demonstrates its performance; Section 5 surveys the existing literature on UWB security and the state-of-the-art on physical layer key generation methods; Section 6 discusses future directions and presents our concluding remarks.

2 SYSTEM DESIGN: OVERVIEW, THREAT MODEL, AND SYSTEM DETAILS

In this section, we first discuss how the reciprocity of wireless channel can be leveraged for generating the secret keys. The potential threat models and the practical challenges facing wireless-channel-based key generation are subsequently introduced. Drawing inspirations from recent advances in machine learning, we then propose a contrastive learning-based key generation scheme that addresses these threats and challenges.

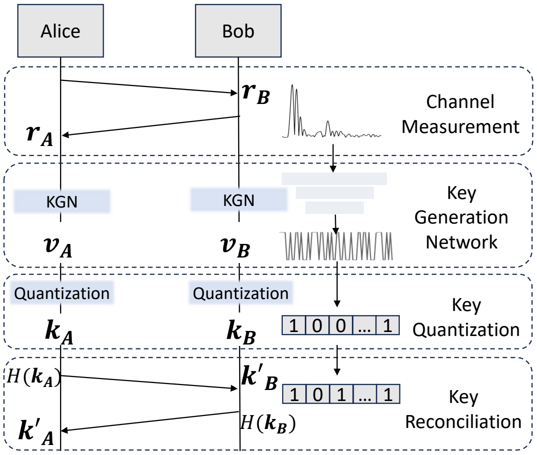


Fig. 2. Illustration of the Key Generation Pipeline

2.1 Secret Key Generation Overview

While symmetric cryptographic methods have demonstrated to be robust solutions for both UWB secure ranging and data encryption and have been adopted in the IEEE 802.15.4z standard, the standard does not specify how the two communicating entities, say, *Alice* and *Bob* agree on the same secret key in the first place. A common approach in the automotive and mobile industries is using an out-of-band (OOB) secure channel, such as Bluetooth, to exchange keys. However, this adds hardware and cost overhead, making UWB reliant on an additional radio technology. For applications like Nearby Interaction or indoor navigation, where users communicate with multiple UWB nodes, pairing with each node is impractical. Alternatively, key exchange protocols like Diffie-Hellman (DH) allow secure key generation over insecure channels but are vulnerable to quantum computing advances.

The third option is to generate the key from a random signal source that Alice and Bob can both observe. One suitable source for key generation is the wireless channel itself, which is reciprocal in a bidirectional communication link, at least in theory. In other words, Alice and Bob should observe an identical set of signal reflections at any time because the signal propagation in the bi-directional link are the same, and therefore they should be able to extract identical keys from the channel impulse response without needing to share any information. In a multipath-rich environment, the wireless channel also decorrelates over space, meaning an eavesdropper, *Eve*, located sufficiently far away from Alice or Bob would observe a different channel and consequently generate a different key.

We now provide an overview of the steps in *UWBKey*'s secret key generation protocol (see Figure 2).

1. **Channel measurement:** Alice and Bob first perform a fast back-and-forth message exchange to measure the impulse response of the channel between them. The turnaround time of the replied message needs to be within the channel coherence time in order for the assumption of channel reciprocity to hold. Typically, a turnaround time of several milliseconds is sufficient for a hand-held device under normal user motions.
2. **Key generation network (KGN):** after Alice and Bob each obtain a CIR measurement, they feed the CIR into a neural network in order to extract the latent features that can help them generate high-quality keys in the next stage. The design of the KGN is detailed in the following section.
3. **Key quantization:** the calculated latent features are subsequently quantized into a binary string by passing through a threshold detector (e.g. a sample larger than the threshold generates a 1 bit and vice versa).

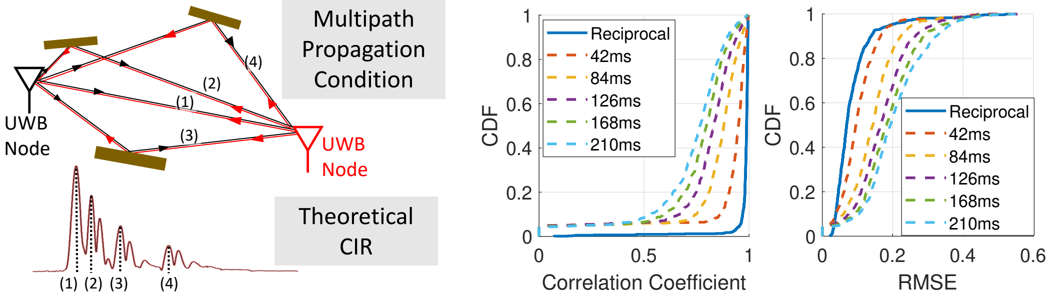


Fig. 3. Illustration and validation of UWB channel reciprocity. Left figure shows two UWB nodes observing identical channel due to the same multipath propagation condition experienced in a bidirectional link. The right figure shows the correlation coefficient and RMSE metric for CIRs measured reciprocally versus measured with varying time gap (specified in legend).

(4) **Key reconciliation:** Due to measurement errors and user movements, the generated keys from the previous stage may not match completely. To reconcile any mismatched keys, forward error correction code can be employed. In this paper, we adopt Reed-Solomon code [25] and BCH code [67] for key reconciliation.

2.2 Threat Model

We consider an adversary who is attempting to obtain the same key as Alice and Bob through passive or active methods. It is assumed the adversary can eavesdrop on the communication between Alice and Bob, but is at least some distance away from either of them. The adversary may also covertly affect the radio propagation conditions, but cannot ever gain full control or complete knowledge of the wireless channel between two arbitrary devices at any arbitrary time. More specifically, we consider the following attacks (see Figure 4) that may be launched against *UWBKey*:

- Passive Attack: In a passive attack, the adversary tries to obtain the secret key by eavesdropping on Alice and Bob. The adversary may also measure the channel impulse response when overhearing any UWB packets.
- Stalking Attack: The adversary may closely follow or stalk a target in hope of observing a more correlated channel as the target. An example would be the attacker placing UWB tags in the target's backpacks, suitcases, or transportation devices that follows the target closely in attempt to gain knowledge of the key.
- Memorization Attack: This attack may work if one of Alice and Bob is stationary. Without loss of generality, assume Alice is an appliance that remains in the same location for an extended period of time. The adversary can record the encrypted messages it overhears, and at the same time note the location of Bob. Later when Bob leaves, the adversary can move to the same location as Bob's previous location where key generation took place, and initiate the key generation message exchange with Alice in attempt to measure the same channel as Bob's previous session.
- Channel Injection Attack: A foreseeable weakness of key generation from the wireless channel is when the channel could be manipulated or predicted by the adversary. For example, by placing a reflector or metasurface [60] that creates a prominent reflection at a certain distance, the adversary could induce a predictable multipath in the CIR measurements made by Alice and Bob. Such a reflector can be placed in advance and hidden to make it difficult to detect.

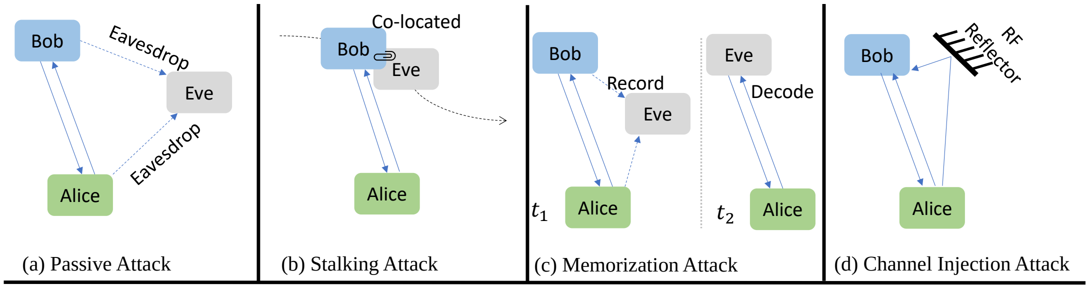


Fig. 4. Threat Model: Different kinds of attacks *UWBKey* should protect against.

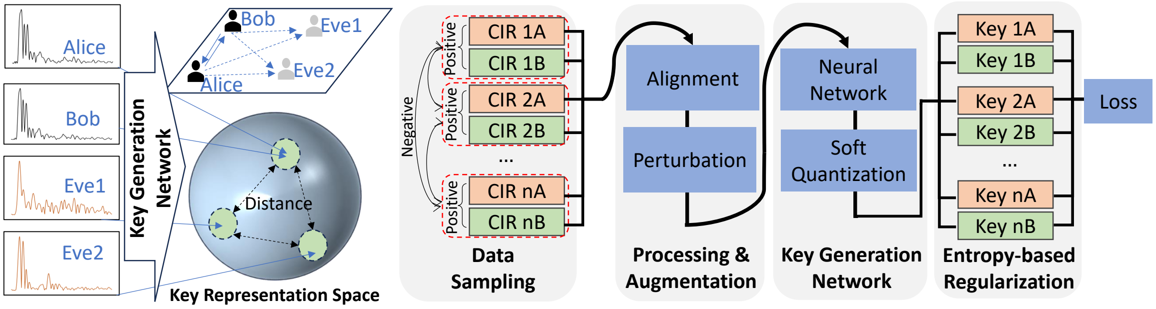


Fig. 5. Illustration of training the contrastive learning-based key generation network.

2.3 Contrastive Learning

We utilize recent advances in contrastive learning to map the raw CIRs to a latent representation that is more suitable for binary key extraction. Contrastive learning is a type of representation learning that learns a feature representation of the data samples based on the notion of being similar or dissimilar to each other. The core idea is to map similar data samples closer together in the representation space while pushing dissimilar points farther apart. This is achieved through two major components: (1) positive-negative sampling, and (2) the contrastive objective function. Taking the classification task as an example, let us suppose the dataset is composed of samples belonging to different class labels. Contrastive learning tries to learn a representation mapping that places samples from the same class (positive samples) closer to each other, while keeping samples from different classes (negative samples) far apart in the representation space. To train a contrastive learning algorithm, first positive and negative samples are randomly drawn based on their class labels, and the representation vector of each sample is computed. In the representation space, we then push for the distance between the positive samples to be small and the distance between the negative samples to be large using the appropriate contrastive objective function.

We observe a striking similarity between the objective of key generation and that of contrastive learning: Alice and Bob wish to agree on the identical secret key by generating binary sequences with minimal bit flip distance, while any Eve located away from Alice or Bob should generate sequences that are dissimilar. Therefore, it is natural to re-formulate the key generation problem as a contrastive learning task *by analogizing the secret key as a latent representation of the wireless channel measurement.* The reciprocally measured CIRs (between Alice and Bob) constitute the positive samples and the CIRs measured at other locations with sufficient spatial separation from Alice and Bob constitute negative samples. In the representation space (or key space), we can similarly push the key distance to be smaller or larger for positive and negative samples respectively (see Figure 5) with carefully designed sampling strategy and objective functions. We will describe this process in more detail in Section 2.4.

It is worth noting that the training process is inherently self-supervised, as it does not require explicit labeling during data collection. To collect the data and form the positive and negative samples, we use two UWB transceivers, one stationary and another moving, that continuously perform fast packet exchange for CIR measurement. Packet sequence IDs and the RX timestamps are recorded alongside the CIRs to be used later as the indicator for the channel probing interval. During data collection, the mobile UWB transceiver moves along a random, non-repeating, sweeping trajectory within the area while CIRs are recorded. The data acquisition process is illustrated in Figure 7(b). To ensure CIR measurement quality, a signal-to-noise ratio (SNR) threshold is applied to filter out low-quality CIR measurements. The positive samples are constructed from the CIR reciprocally measured between the two UWB transceivers with a delay of a few milliseconds. The negative samples, on the other hand, are derived from CIRs corresponding to incoherent channels with distinct multipath propagation conditions. Since the CIRs are measured along a random and non-repeating trajectory, negative samples can be obtained by simply selecting CIRs measured at different times using the RX timestamp information.

2.4 Objective Function

We adapt the normalized temperature-scaled cross entropy loss (NT-Xent loss) function [15] as the objective for training the contrastive learning algorithm. As illustrated in Figure 5, each time a minibatch of N pairs of reciprocally measured CIRs is randomly sampled from the training data set. For every sample $\mathbf{v}_i$, there is one positive sample $\mathbf{v}_i^+$, and $(N-1)$ negative samples $\{\mathbf{v}_{j|j=1,\ldots,N,j\neq i}\}$. The loss function for the minibatch is defined as:

$$\mathcal{L}_{Contrastive} = -\frac{1}{N}\sum_{i=1}^{N} log\frac{exp(sim(\mathbf{v}_i, \mathbf{v}_i^+)/\tau)}{\sum_{j=1}^{N} \mathbb{1}_{j\neq i} exp(sim(\mathbf{v}_i, \mathbf{v}_j)/\tau)},$$

where $sim(\cdot)$ is the similarity function between two representation vectors, τ is the temperature parameter, and $\mathbb{1}_{j\neq i}$ is the indicator function that only evaluates as 1 if $j \neq i$, and as 0 otherwise.

Although the ultimate objective of contrastive learning is to generate binary keys that are identical for a positive input pair and random for negative input pairs, generating binary numbers from continuous inputs entails a quantization process, which does not have a usable gradient for training. Additionally, the similarity metrics between binary vectors, such as the hamming distance, is also a non-differentiable function. Therefore, we propose to use continuous, differentiable functions with non-zero gradient to approximate the quantization process and hamming distance function. The soft quantization function we use is:

$$\mathbf{v_i} = SoftQuant(\mathbf{z}_i) = \tanh(\alpha \mathbf{z}_i),$$

where α is a hyper-parameter that controls the shape of the soft quantization function.

Similarly, hamming distance is based on counting the number of bit flips, which is a non-differentiable function. As a continuous function alternative, we propose to use the length-normalized dot product as a measure of similarity between two vectors:

$$sim(\mathbf{v}_i, \mathbf{v}_j) = \max\left(0, \frac{\mathbf{v}_i^T \mathbf{v}_j}{M}\right),$$

where M is the key length. Given that $\mathbf{v}_i$ and $\mathbf{v}_j$ are output from the chosen soft quantization function that produces output values close to -1 and 1, the distance function is equal to 1 if $\mathbf{v}_i$ and $\mathbf{v}_j$ are identical, and 0 if there is a 50% mismatch. (The zero-comparison operator is used in order not to push the similarity metric to the negative region, which would indicate negative correlation and therefore lower entropy once again).

2.5 Max Entropy Regularization

Apart from satisfying the vector similarity-dissimilarity objective, randomness of the secret key is one of the most important design considerations in key generation schemes. In fact, any correlation or predictable structure in key generation will make it easier for Eve to produce the same key. Entropy, the most commonly used metric for randomness, is defined as:

$$H(\mathbf{v}) = -\sum_{\mathbf{v} \in \mathbf{V}} p(\mathbf{v}) \log p(\mathbf{v})$$

where $p(\mathbf{v})$ describes the marginal probability distribution of a discrete random variable $\mathbf{v}$. Intuitively, we can obtain an embedding with maximal randomness by simply maximizing $H(\mathbf{v})$ as part of the loss function. However, this does not work for a couple of reasons: (1) To evaluate $H(\mathbf{v})$, we first need to obtain the histogram of $\mathbf{v}$ to estimate its probability distribution $p(\mathbf{v})$. Since there is a counting process involved and counting is not a differentiable function, this method of estimating $H(\mathbf{v})$ cannot be used in gradient descent-based learning algorithms. (2) For high-dimensional $\mathbf{v}$ vectors (in our case the dimension is the key length), an enormous amount of data and computation is needed to estimate its marginal distribution, making this method intractable. Therefore, the question is: *is there a function that can estimate the entropy with a limited number of data and is also differentiable?*

To solve this problem, we use the Maximum Entropy Encoding (MEC) approach [41, 44], which uses minimal coding length as a proxy measure for entropy, since entropy can be interpreted as the number of bits needed to encode information. The MEC function is given by:

$$\mathcal{L}_{MEC} = -\frac{N+M}{2} log \left(det \left(\mathbf{I}_N + \frac{M}{N\epsilon^2} \mathbf{V}_1^T \mathbf{V}_2 \right) \right) \approx -Tr \left(\mu \sum_{k=1}^{K} \frac{(-1)^{k+1}}{k} (\lambda \mathbf{V}_1^T \mathbf{V}_2)^k \right),$$

where N is the batch size, M is the dimension of the embedding vector, $\mathbf{V}_1 = [\mathbf{v}_1^1, \mathbf{v}_1^2, ..., \mathbf{v}_1^N]$ contains the embedding vectors of one batch, $\mathbf{V}_2 = [\mathbf{v}_2^1, \mathbf{v}_2^2, ..., \mathbf{v}_2^N]$ contains the reciprocal version of $\mathbf{V}_1$, $\lambda = \frac{M}{N\epsilon^2}$, and $\mu = \frac{N+M}{2}$. Note that the approximation on the second line is the result of keeping the first K significant terms in the Taylor expansion of the first line for simplifying the determinant calculation.

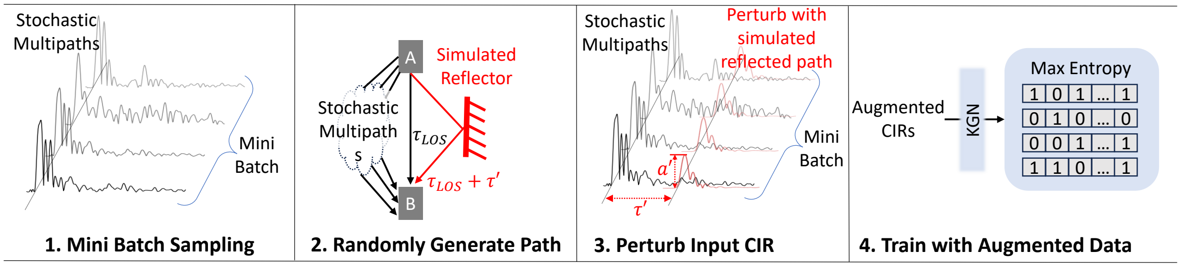
We incorporate the MEC loss as a regularization term on top of the contrastive loss. In other words, the composite loss function can be expressed as:

$$\mathcal{L} = \mathcal{L}_{Contrastive} + \beta \mathcal{L}_{MEC},$$

where β is a hyper-parameter that determines the weight of the regularization term.

2.6 Data Augmentation to Counter Channel Injection Attacks

So far, we have discussed utilizing contrastive learning to learn an embedding of the CIR which maximizes certain similarity-dissimilarity and entropy metrics. However, we have not addressed an important weakness of channel-based key generation: *what if the adversary can manipulate the wireless channel itself to force a particular key pattern at Alice and Bob?* Of course, achieving full control of wireless channel is extremely difficult even in meticulously set up environments. However, it may be possible for the adversary to introduce certain deterministic patterns into the wireless channel by placing RF reflectors close to Alice and Bob. For example, if one of Alice and Bob is a public appliance, the adversary could hide such RF reflectors at specific distances nearby, and when key generation takes place, the CIR measurements may contain reflections that are known or easily predictable



1. Mini Batch Sampling
2. Randomly Generate Path
3. Perturb Input CIR
4. Train with Augmented Data

Fig. 6. Illustration of data augmentation to counter channel injection attacks.

by the adversary. In face of this problem, we need to make sure that the embedding should still be unpredictable even under partially controlled radio propagation conditions.

The core idea is, since there are still other multipaths from other objects in the environment that are not controlled by the adversary, then our key generation network should try to extract information from a wide range of input features. In other words, the output representation should not be strongly correlated with just a few select taps in the input CIR.

We propose to use data augmentation to solve this problem. During training, we artificially introduce strong reflections to the CIR data to simulate a channel forcing attack. To achieve this, we first measure a template signal $p(t)$ that is free of any multipath by connecting two UWB transceivers through a coaxial cable. Then, any measured CIR can be modeled as:

$$r(t) = \sum_{l=1}^{L} a_l e^{-j2\pi f_c \tau_l} p(t - \tau_l), \tag{1}$$

where a_l, τ_l are the amplitude and delay of the l^{th} path respectively. We introduce perturbation to the measured CIR by adding additional path following the same model:

$$r^{aug}(t) = \sum_{l=1}^{L} a_l e^{-j2\pi f_c \tau_l} p(t - \tau_l) + a' e^{-j2\pi f_c \tau'} p(t - \tau'), \tag{2}$$

where a' and τ' are artificially tuned to control the the amplitude and delay of the artificially introduced path. The augmentation is applied to a whole batch of training data and the MEC regularization is applied. Therefore, the model is forced to still generate the representation with maximum entropy while all the input contain a common perturbation. This process is illustrated in Figure 6, and explained below:

(1) During the training phase, every time we draw a batch of samples (as explained in Section 2.3), the sampled CIRs are free of deterministic components as they are randomly drawn from the training dataset and they only contain the multipaths resulting from the varying signal propagation environment.
(2) We generate a reflected path with randomly generated delay τ' and magnitude a' to simulate a channel injection attack. Note that this path is randomly generated at every training iteration for every mini batch such that a diverse range of path delay and magnitude may be seen by the training algorithm. a' is selected from a uniform distribution between 0 and 1 (with 1 representing the generated path having the same magnitude as the first path). τ' is selected from a uniform distribution between 0 ns and 80 ns, which is the full range of CIR we use for key generation.

(3) The CIR contribution of this artificially generated path is subsequently added to all CIR samples within the mini batch using Equation (2), as if all measurements experience a deterministic reflected path at a certain distance with a certain signal strength.
(4) After data augmentation, the perturbed CIR samples are treated as normal input samples to the training algorithm. The intuition is: the MEC regularization will make the network learn to increase the entropy of the generated keys without overly relying on any specific CIR tap and become resilient against channel injections.

In addition, we use a random number generator to randomly decide whether data augmentation is performed or not for each mini batch. This is to make sure the model not only sees artificially generated data, but also the originally measured data during training. In our final implementation, there is a 30% chance a batch gets augmented.

In summary, we utilize machine learning techniques coupled with significant domain knowledge to develop a function that converts raw channel impulse response obtained by two UWB devices communicating with each other, into robust symmetric keys. We focus on entropy maximization and on keeping the keys robust despite multipath maliciously introduced by an adversary to try to increase predictability. Next, we describe our real-world prototype implementation, followed by evaluation of the system.

3 IMPLEMENTATION

The *UWBKey* system is implemented on commercial-off-the-shelf (COTS) UWB hardware from Decawave (now Qorvo). The UWB communication and channel measurement logic are implemented on a set of UWB transceivers based on Decawave DW1000 chipsets, which are compliant with the IEEE 802.15.4z standard and use standard UWB bands as allowed by the FCC. During the operation of *UWBKey*, we ensured we do not have other UWB transmissions in the vicinity, which allowed us a controlled environment for reliable experimentation. Wi-Fi and Bluetooth signals were present throughout our tests but do not significantly interfere with the UWB signals. In all of the following experiments, we used the Channel 5 with a center frequency of 6.4896 GHz, preamble length 128 and data-rate of 6.8 Mbps. An HP Envy x360 laptop was used to collect CIR data and train the neural network for contrastive learning-based key generation network. The data will be open sourced.

3.1 Baseline Methods

To understand how well our proposed method perform compared to existing methods, we implement the following quantization methods as baselines.
Noise Adaptive Quantization (NAQ) In [34], the authors proposed a magnitude-based adaptive quantization scheme. The key idea is to only use samples with significant magnitude for quantization and the samples with low SNR will be discarded by setting a guard band. Instead of setting a fixed guard band threshold, the channel measurement is divided into smaller windows, and in each window the guard band boundaries are determined adaptively to the samples' standard deviation: $q^{+} = mean - \alpha * std$, $q^{-} = mean - \alpha * std$. The samples larger than q^{+} are quantized to 1 and the samples smaller than q^{-} to 0, repeated until the desired key length has been reached. Here, we choose $alpha = 0.1$ and a window size of 10 ns. We consider this method the closest to raw CIR without relying on pre-defined threshold.
Gradient-based Quantization (GBQ) The authors in [37] describe a channel quantization scheme based on the change of the measurement. An increase in the measurement is quantized as 1 and a decrease is quantized as 0.
Siamese Network (SN) The authors of Blind Twins [57] were the first to leverage contrastive learning for secret key generation. A Siamese Network is trained to find the representation such that the percentage of bit mismatch from reciprocal CIRs are "pushed" to 0 and to 50% for eavesdroppers. For fair comparison, the same CNN neural

network architecture is used for both Siamese Network and *UWBKey*. Our system benefits from the max entropy and data augmentation which is not used in BlindTwins [57].

CIR-DFT Magnitude with Savitzky-Golay Filter (DFT+SGF) In [19], the authors propose to first extract the middle section of the measured CIR which contains significant multipaths, take the discrete Fourier transform (DFT), and then only keep the magnitude of the DFT result. The reasoning of this operation is to mitigate the mismatch due to false alignment of the measured CIRs, since misalignment in the time domain translates to a phase shift in the frequency domain, which gets ignored by only taking the magnitude response. The author also suggests truncating the DFT result to keep only the significant frequency range prior to quantization.

CIR-DFT Magnitude with Multilevel Quantization (DFT+M-Quant) In [9], the authors also propose taking the DFT of the measured CIR (magnitude only) for similar reasons. In addition, the authors propose to use an adaptive multilevel quantization scheme [62], which works as follows. First, the signal is segmented into several non-overlapping windows. For each window, the number of quantization levels is decided based on the estimated entropy of the present window. Finally, the signal is quantized into bits based on the number quantization levels for every window determined in the previous step.

3.2 *UWBKey* Implementations

We implement two versions of *UWBKey*: a vanilla version of *UWBKey* without using MEC regularization or data augmentation, and the full version of *UWBKey*.

***UWBKey*-Vanilla** In contrast to siamese network that only contrasts two samples at a time, we train *UWBKey*-Vanilla by comparing each sample with its reciprocal pair and against multiple randomly drawn negative samples. A CNN is chosen as the contrastive learning architecture, which is composed of four convolutional blocks followed by four fully connected layers (our optimizations for on-device deployment are shown in Figure 14). Each convolutional block consists of a convolutional layer, batch normalization, ReLu activation, and max pooling layer, and to prevent overfitting, a dropout is used before the output layer.

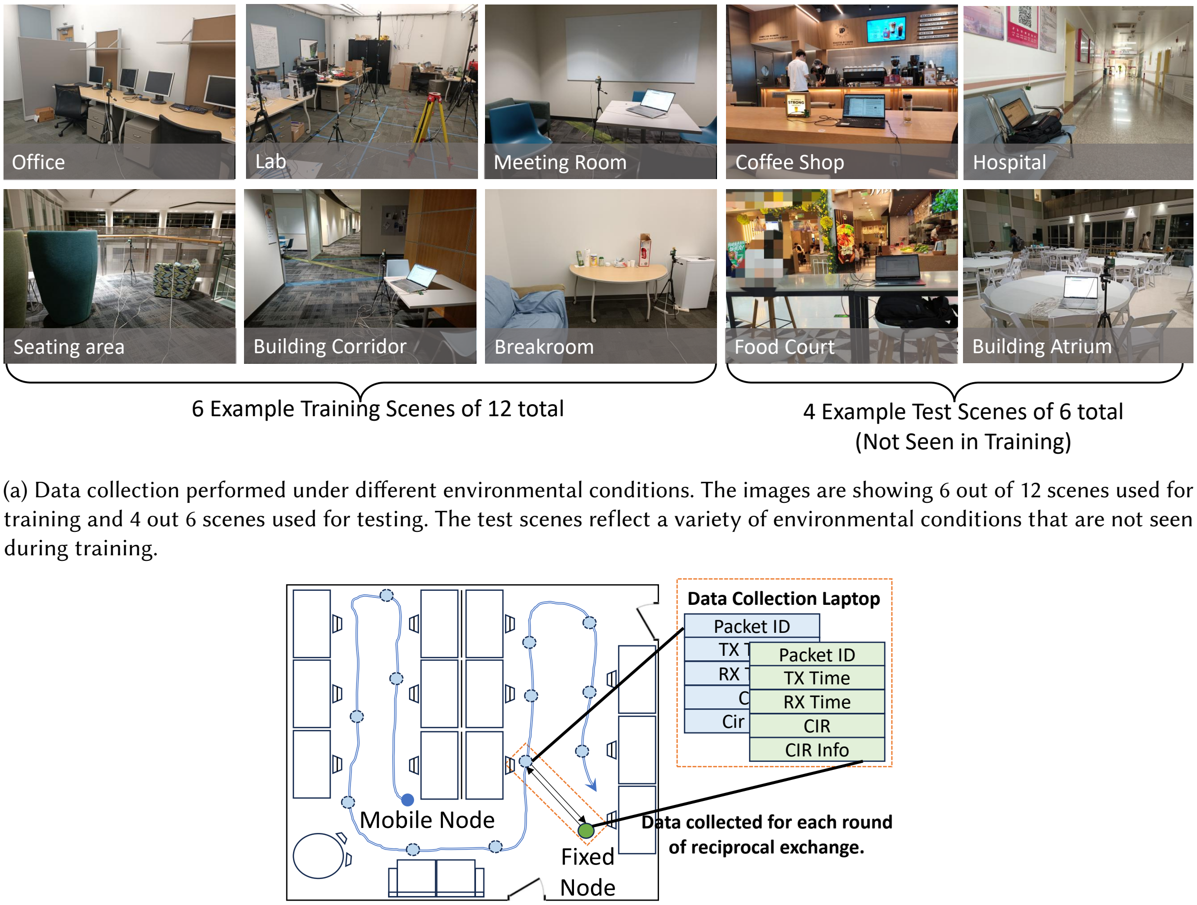
***UWBKey*-Full** In addition to *UWBKey*-Vanilla, we introduce Max Entropy regularization to enforce randomness in the learned representation. To further ensure that the representation does not have strong correlations to any local perturbation in the CIR measurement (such as a strong reflected path introduced by an adversary to force a particular pattern in the observed channel), we employ data augmentation to simulate channel injection attacks and help the trained network to be robust against these attacks. This represents the full form of *UWBKey* and we will demonstrate how it compares against the other baseline methods in the next sections.

3.3 Hyperparameters

UWBKey is trained to take in the first 80 ns portion of the CIR (real and imaginary) starting from the first path and output a binary key with a length of 128 bits as prescribed by the IEEE 802.15.4z standard for appropriate security. The system hyperparameters are chosen by optimizing the validation loss. We found the optimal set of hyperparameters to be as follows. The scaling factor α in the soft quantization is set to 15. The weight given to the MEC regularization loss β in Section 2.5 is set to 0.4. During training, data augmentation is applied stochastically to the input with a probability of 0.3. The injected path in data augmentation has a randomly generated delay τ' uniformly drawn between $[0, 80]$ ns, and amplitude a' uniformly drawn between $[0, 1]$ (times the amplitude of the first path, assuming that the first path is still one of the strongest path due to device proximity). The batch size is set to 16. These hyperparameters are used throughout the remaining paper.

4 EVALUATION

In this section, we evaluate the proposed system under different metrics and scenarios.



(a) Data collection performed under different environmental conditions. The images are showing 6 out of 12 scenes used for training and 4 out 6 scenes used for testing. The test scenes reflect a variety of environmental conditions that are not seen during training.

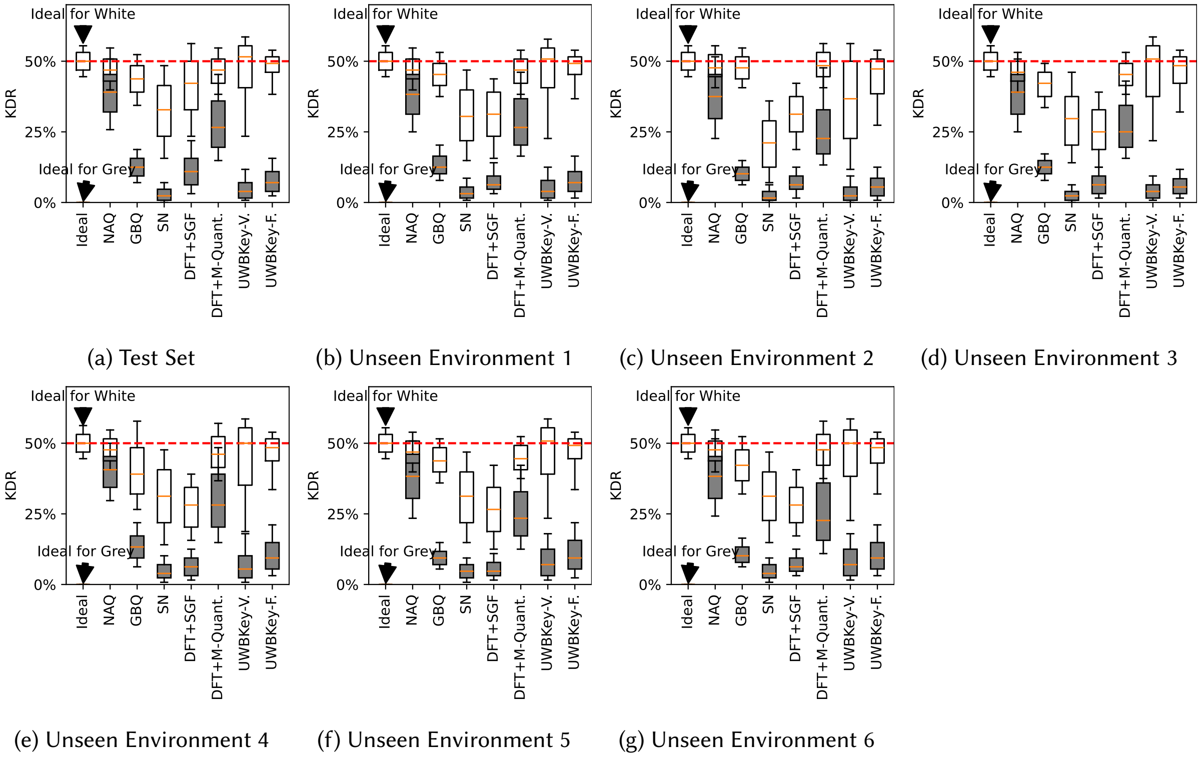
(b) Illustration of data collection setup with one fixed node and one mobile node with a random, non-repeating trajectory. The packet and CIR information are collected on a laptop.

Fig. 7. Illustrations of some of the data collection scenes and data collection methodology.

4.1 Dataset

For data acquisition, we use a pair of UWB transceivers that perform continuous back-and-forth message exchange at around 32 Hz rate as shown in Figure 2. During data collection, we fix one UWB transceiver to stay stationary at one location while the other is constantly moved around in a random trajectory, such that the CIRs measured at different time instances can be treated as spatially separated due to different multipath propagation conditions. For every packet, the packet ID, TX/RX timestamps, CIR and other RX information, such as the sample index of the first path and the received signal power, are recorded. Figure 7(b) shows an illustration of the data acquisition setup.

To ensure that the trained model does not overfit to a particular environment, we repeat the measurements in 12 different physical locations (which we call scenes), with roughly 2500 CIR samples in each scene. The different scenes are meant to cover a variety of common operating environments where secure communication



(a) Test Set (b) Unseen Environment 1 (c) Unseen Environment 2 (d) Unseen Environment 3

(e) Unseen Environment 4 (f) Unseen Environment 5 (g) Unseen Environment 6

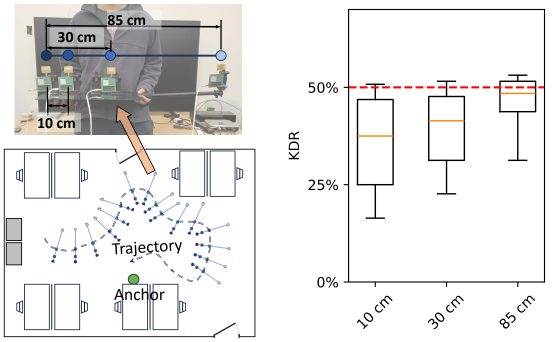
Fig. 8. Distribution of Key Disagreement Rate of Positive and Negative CIR Pairs with Data from the Test Set and 6 Unseen Environments. Grey boxes indicate positive pairs. White boxes indicate negative pairs. For reference, the KDR distribution of the ideal key generation method is shown on the left (KDR of the positive pairs are always 0).

may be useful, including 6 scenes in residential areas (2 bedrooms, a dining room, a living room, a kitchen, and a corridor), and 6 scenes in public buildings (2 offices, 2 hallways, a laboratory, and a breakroom). The data is then split 90 − 10, with 90% used for training and the rest for testing. We collect data in 6 additional scenes: an outdoor plaza, a building atrium, an academic office, a coffee shop, a foodcourt, and a hospital (denoted as unseen environment 1-6), withheld from training and only used in testing in order to evaluate the generalization capability of the learned model to unseen data. Figure 7(a) shows the photos of a few scenes where CIRs have been recorded. All CIRs are time-aligned with respect to the first path based on the reported first-path index from DW1000, and the amplitude and phase are normalized with respect to the first path as well. **In total, we have analyzed close to 48,000 CIRs. All of the collected data will be made open to support the research community.**

4.2 Key Disagreement Rate

As explained previously, an ideal key generation scheme should ensure that (1) Alice and Bob generate keys that are identical; (2) for CIRs measurements that are spatially/temporally separated, the generated binary keys are random and independent with maximum entropy. The key disagreement rate (KDR) measures the percentage of inverted bits between two binary keys and is defined as the ratio between the hamming distance and the key length. For reciprocal pairs, since they need to generate identical keys, the key generation scheme should achieve

KDR close to zero. On the other hand, for CIRs measurements that are spatially/temporally separated, an ideal key generation scheme should produce keys that are independent and random with equal probability of 1's and 0's. It can be derived that the hamming distance should follow a binomial distribution with a mean of half of the key length, implying that the KDR also follows a scaled binomial distribution centered at 0.5. Therefore, we can examine whether the generated keys follow the desired properties by *comparing the KDR distribution of the generated keys against the ideal distribution.*



(a) Illustration of the experimental setup of spatial variation test with mobile tags.

(b) Distribution of KDR with varying spatial separation.

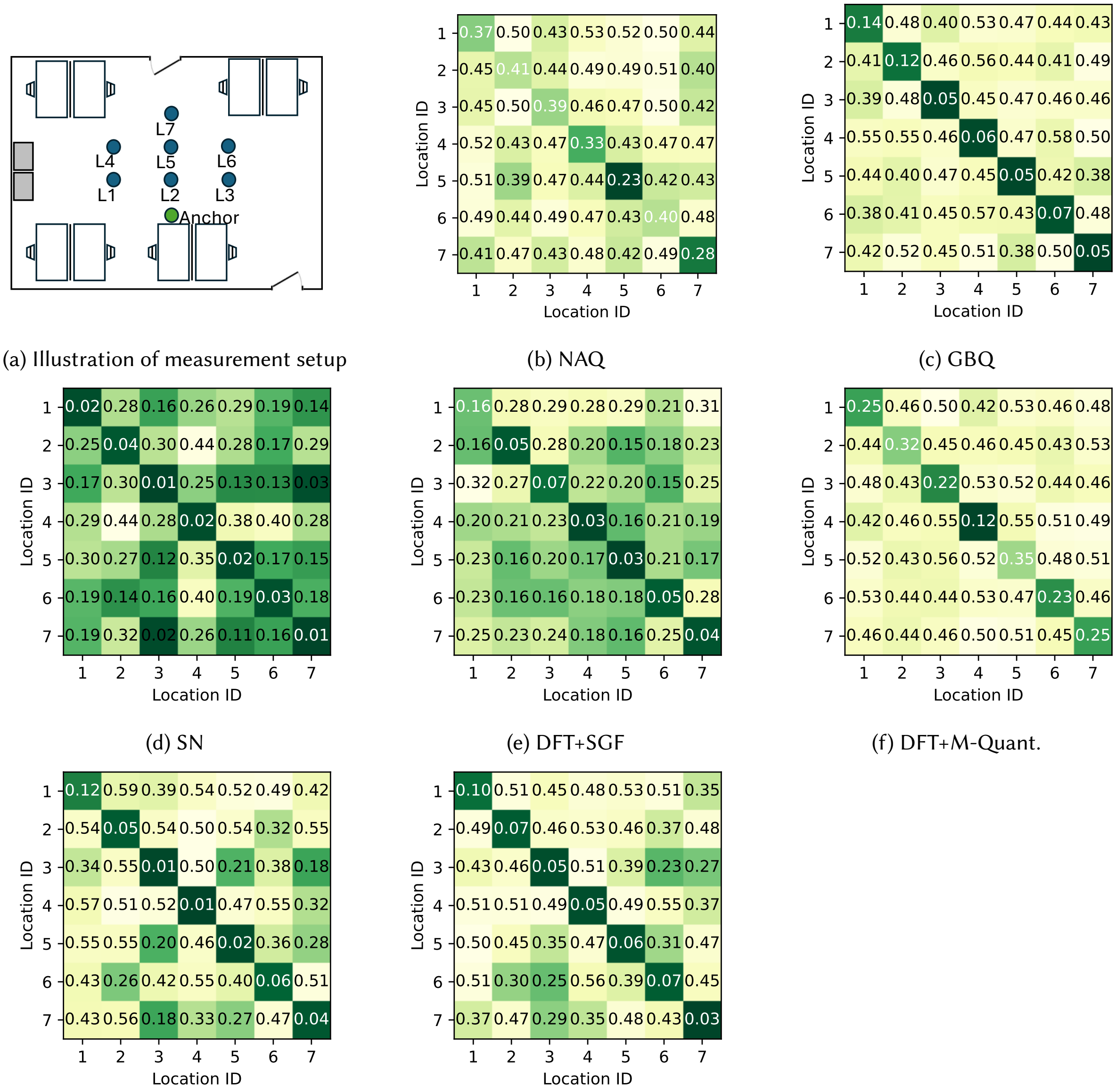
Fig. 10. Investigation of key spatial variation for mobile tags. The setup of four UWB tags fixed on a rigid body moved in a random trajectory is shown in (a). The KDR between tags with varying separation distances is shown in (b).

First, we validate the KDR distribution of *UWBKey* and compare it with the baseline methods using the test dataset. Note that the test dataset is not observed during training, but was collected in the same environments as the training data. The KDR distribution of the generated keys is shown in Figure 8(a), with the box representing the $25th-75th$ percentile range, and the end whiskers representing the $10th$ and $90th$ percentile of the KDR. Grey shading is used for the positive pairs generated from reciprocal CIR measurements while white shading is for the negative pairs. The ideal KDR distribution, assuming the keys are independent and random with equal probability of 1's and 0's, is shown on the left for reference. We observe that for positive pairs, the contrastive learning-based methods achieve lower KDR compared to directly quantizing the CIR. This demonstrates that the trained neural network is capable of extracting latent features that are closely matching despite the noise and measurement errors in the CIR. DFT+SGF method also displays low KDR for positive pairs, most likely contributing to the Savitzky-Golay filter which reduces the effect of noise. Among the contrastive learning-based approaches, SN also achieves the lowest bit error rate. However, for negative pairs, neither DFT+SGF nor SN method could push the KDR distribution towards 50%, meaning the keys have low entropy. DFT+M-Quant method pushes the KDR distribution for negative samples to be similar towards the ideal distribution, indicating the generated keys have high entropy. On the other hand, it produces poor reciprocity for positive pairs, which is far from ideal. It can be observed that *UWBKey*-Full is the closest match to the ideal distribution, which most likely attributes to the MEC regularization. Without entropy-based regularization, we can see both SN and *UWBKey*-Vanilla produce a skewed distribution with negative samples performing poorly.

To ensure the result generalizes to the data collected in all environments, we examine the KDR of the generated keys in **6 unseen environments** withheld from training (see Figure 8(b)–(d)). We can observe that results are very similar to that of the test data. From this result alone, *UWBKey*-Full is the best performer as it achieves fairly low key errors for legitimate parties while maintaining high entropy.

4.3 Spatial Variation

One important requirement that the generated keys should satisfy is spatial variation—any device located at a different location should not be able to generate the same key as the legitimate user. Therefore, it is important to test that there is significant mismatch in the secret keys generated from CIRs measured at different locations. We set up this test such that no specific location receives any preferential treatment. We set up one anchor and 7



(a) Illustration of measurement setup

(b) NAQ

(c) GBQ

(d) SN

(e) DFT+SGF

(f) DFT+M-Quant.

(g) *UWBKey*-Vanilla

(h) *UWBKey*-Full

Fig. 9. Key Disagreement Rate with spatially separated nodes. For diagonal entries, the smaller the value the better. For other entries, values closer to 0.5 are better.

tags placed at different locations in a room. Figure 9(a) shows an illustration of the setup. The distance between adjacent tags is set at 1 m. All 7 tags perform message exchanges with the anchor for CIR measurements in a time-divided fashion, and while one tag is communicating with the anchor, all the other tags act as eavesdroppers who are also measuring the CIR with respect to the anchor to simulate attackers launching passive attacks from multiple spatial locations. The CIRs are fed into the key generation network and the overall KDR is calculated between all 7 tags. Figure 9(b)–(f) show the average KDR between the 7 tag locations under different methods. First of all, the diagonal entries represent KDR between the reciprocal CIR pairs measured by the tag and the anchor, and therefore the KDR should be close to zero. All methods except NAQ and DFT+M-Quant achieve low bit error rate. SN achieves especially low KDR with all 7 tags (less than 4%). Second, the off-diagonal elements represent the KDR between Alice/Bob and Eve in the vicinity and the values should be centered around 50%. SN and DFT+SGF produce KDR significantly lower, indicating that an eavesdropper has a higher probability of predicting the generated key. Comparing *UWBKey*-Vanilla and *UWBKey*-Full, *UWBKey*-Vanilla achieves very low KDR for legitimate parties while for certain eavesdropper locations does not produce enough separation (the lowest bing 18% only between Location 3 and 7). On the other hand, *UWBKey*-Full is more balanced, achieving lower than 10% bit error rate for legitimate parties across the board while ensuring sufficient key separations from eavesdroppers. Together with the result in Figure 8, it is reasonable to deduce that adding the entropy-based regularization trades off the key error rate for a representation with higher randomness.

We also fix four UWB receivers on a rigid body to investigate the key spatial variation in the mobile scenario. One UWB is treated as the legitimate user and performs key generation exchange with one fixed anchor node, and the other three receivers are treated as eavesdroppers and are separated by 10 cm, 30 cm, and 85 cm from the first UWB respectively. When in motion, the four receivers stay in the same relative positions to each other (see Figure 10(a)). Note that this also simulates a passive stalking attack, in which an adversary follows the legitimate user closely at all times, or sticks a UWB tag to the user's luggage, etc. in an attempt to gain knowledge of the secret key. Figure 10 (b) shows the KDR distribution between the three eavesdroppers' keys and the legitimate user's key. With 10 cm device separation, the KDR distribution is strongly skewed towards 0. As the device separation increases, the eavesdropper's keys show stronger decorrelation, and at 85 cm the KDR distribution is approaching the ideal key generation scheme, meaning no correlation with the user's generated key. The results in Figure 9 and Figure 10 suggest high effectiveness against passive attacks by both static and stalking adversaries located 85 cm and beyond from Alice or Bob.

4.4 Randomness

We further test the randomness of the binary keys generated in the unseen environments using the NIST Statistical Testing Suite for Random Number Generator [6], which is the standard toolbox for evaluating the quality of random number generators. The randomness is assessed based on multiple hypothesis testing criteria, such as checking for long runs of zeros and ones. Table 1 summarizes the test results with the associated p-values for different quantization methods. The p-values range between 0 and 1, representing the probability of the observation under the null hypothesis of each test. A higher value means higher confidence of randomness, and conventionally a value larger than 0.05 is required to pass a test.

We can observe that the methods directly extracting keys from CIR perform poorly. This is likely the result of predictable structures in the CIR itself. While SN and *UWBKey*-Vanilla both pass a few test, *UWBKey*-Full passes all tests with high confidence level. This shows the effectiveness of the MEC regularization in learning a highly random representation.

4.5 Channel Injection

In this section, we investigate how robust *UWBKey* is when facing a channel injection attack.

First, we check the input-output correlation of the key generation network to ensure the representation does not have strong correlation with just a few select input features. Otherwise, the adversary could launch a channel injection attack by introducing predictable reflected paths in the CIR to try to force a semi-deterministic output

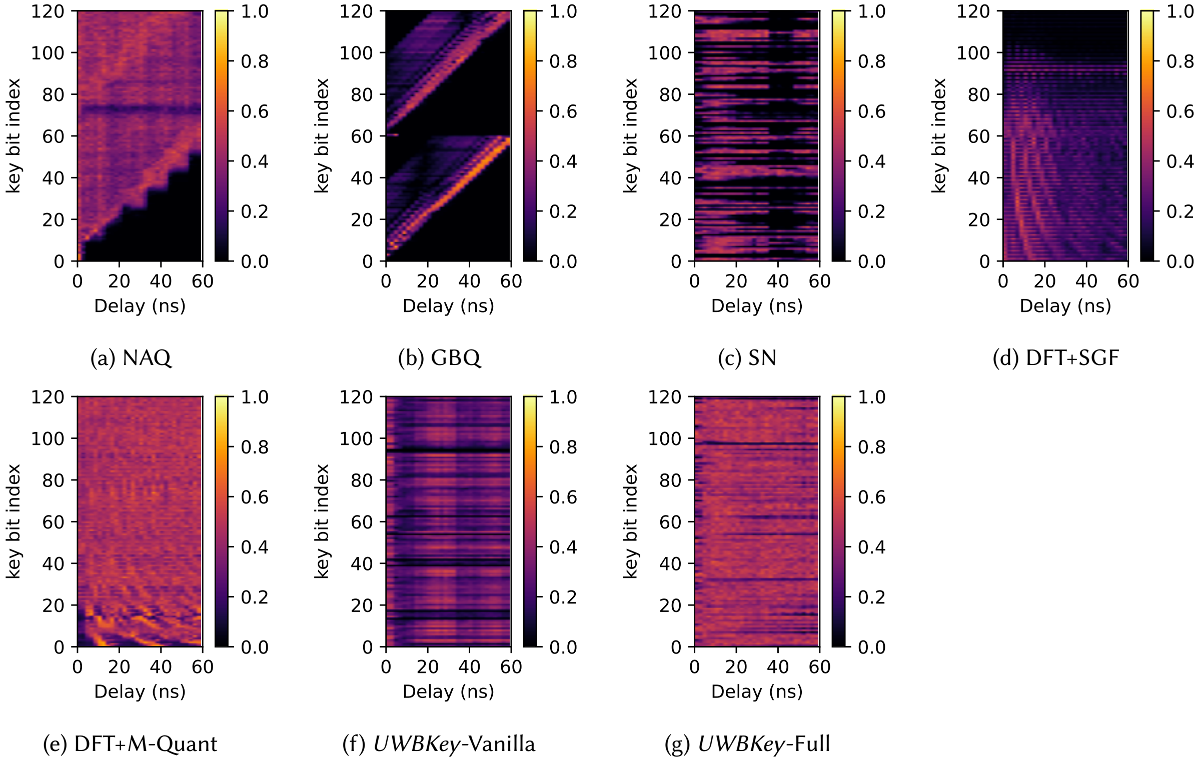
Table 1. NIST Statistical Test for Random Number Testing [6]. The p-value in the range of $[0, 1]$ is shown for each test. A p-value greater than 0.05 is bolded to indicate a passed test. The test names are listed as follows: RT1-Monobit, RT2-Block Frequency, RT3-Run of Ones, RT4-Longest Run of Ones, RT5-Non-overlapping Pattern, RT6-Overlap Template Matching, RT7-Linear Complexity, RT8-Cumulative Sums (Forward), RT9-Cumulative Sums (Backward).

Type of Test	NAQ	GBQ	SN	DFT+SGF	DFT+M-Quant	*UWBKey*-Vanilla	*UWBKey*-Full
RT1	**0.4061**	0.0000	0.0000	**1.0000**	**0.2293**	**0.7370**	**0.0801**
RT2	**1.0000**	0.0000	**0.9999**	**1.0000**	0.0027	0.0135	**0.9341**
RT3	0.0000	0.0000	0.0000	0.0216	0.0000	**0.6721**	**0.8255**
RT4	0.0000	0.0000	0.0110	0.0000	0.0000	**0.2620**	**0.9981**
RT5	0.0000	0.0000	**0.2520**	0.0000	0.0000	0.0040	**0.3294**
RT6	**0.1507**	0.0000	**0.6539**	0.0000	0.0000	**0.4023**	**0.8722**
RT7	**0.4645**	0.0140	**0.1690**	**0.7986**	**0.3674**	**0.9003**	**0.2870**
RT8	**0.6303**	0.0000	0.0001	**1.0000**	**0.2993**	**0.6712**	**0.1037**
RT9	**0.6303**	0.0000	0.0001	**1.0000**	**0.1294**	**0.7043**	**0.0708**

key. We ask the question: how does an artificially introduced path affect the generated key? To answer this question, we artificially introduce perturbations in the CIR and observe how the output behaves. Here, we introduce a variable path using the same method described in Equation (2), where we vary the path delay between 0 and 60 ns. And without loss of generality and for clarity of analysis, the path amplitude is chosen to be 0.8 times the amplitude of the first path (in a subsequent experiment we relax this assumption). Then, we add this artificially generated path to all CIR measurements from a batch of the dataset from the unseen environments, and observe which bit indices are more likely to flip in the generated key.

Figure 11 shows a heatmap of the bit flipping probability for various delay associated with the added path. For example, we can observe that GBQ has a $\sim$ 70% of chance of bit flipping around bit index 20 and 80 when the added path has a delay of 20 taps. (Since we split the CIR into real and imaginary part and concatenate them before quantization, the keys also flip at two places corresponding to the perturbed real and imaginary parts.) This can be explained by GBQ directly quantizing the CIR, causing a high correlation between the location of the CIR perturbation and the location of the bit flips. We can observe that NAQ and GBQ both experience very localized perturbations in the generated keys corresponding to the delay of the added path. This means that an adversary can force bit changes at certain indices by placing strong reflectors near Alice and Bob, which greatly reduces the security. In SN and *UWBKey*-Vanilla, we can see that there is no longer a strong predictable pattern between input and output perturbations, but the location of the input perturbation is still strongly correlated with several output indices, which is not ideal. In *UWBKey*-Full however, we can observe that an added path with any delay induces a 50% chance of bit flipping in all key indices, indicating that any input feature has roughly equal contribution as any other input feature in determining the output.

To understand how data augmentation combined with MEC regularization helped the model learn a more robust representation against channel injection attacks, we perform an ablation study by comparing *UWBKey*-Full and *UWBKey*-Vanilla. We perturb all CIRs measured in the unseen environment by adding an artificial path with the variable amplitude ($a' = 0.3, 0.8, 1.5$) and delay ($\tau' = 20, 30, 40, 50, 60ns$). This essentially simulates a channel injection attack where a reflected path is always present. Then we examine the randomness of the generated keys using the NIST Test Suite. The number of passed randomness tests are shown in Figure 12, where the full results of the p-values for each test can be found in Section A. When a weaker injected path is present ($a' = 0.3$), *UWBKey*-Full still passes all or most of the tests with high p-score. As the injected path grows stronger ($a' = 0.8$), the randomness is reduced, but still at least 5 out 9 tests are passed. When the injected path becomes even



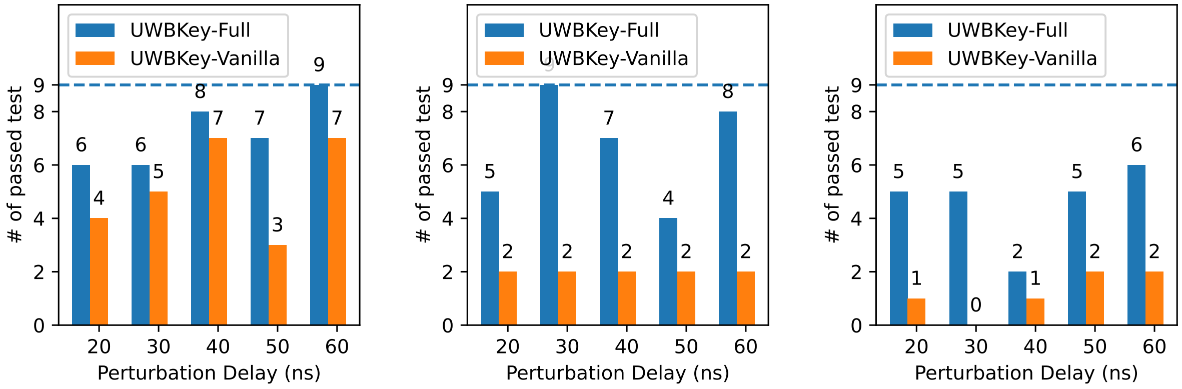
(a) NAQ
(b) GBQ
(c) SN
(d) DFT+SGF
(e) DFT+M-Quant
(f) *UWBKey*-Vanilla
(g) *UWBKey*-Full

Fig. 11. Visualization of input-output correlation of different quantization scheme with input perturbation. The color of the heatmap shows the probability of bit flip after the injected path is introduced.

stronger than the first path ($a' = 1.5$), fewer tests are passed indicating the generated keys become less random. On the other hand, comparing *UWBKey*-Full and *UWBKey*-Vanilla, we observe that *UWBKey*-Full consistently produces higher randomness scores than *UWBKey*-Vanilla, which we attribute to the data augmentation and MEC regularization used during training. A detailed table showing results of all 9 randomness tests over different a' and τ' values is included in Section A.

4.6 Temporal Variation

Aside from spatial variation, it is also important that the keys generated at the same location change over time. Since CIR is treated as a source of randomness for extracting information, the amount of information contained in a signal is directly related to its entropy. If the channel remains static, a memorization attack can be launched, where the adversary memorizes the encrypted messages temporarily, and later moves to the same location as where the legitimate user was situated to measure an identical channel. Indeed, if both Alice and Bob are stationary and there is no other changes in the environment, *UWBKey* would generate invariant keys over time. We argue, however, in most of the considered use cases, at least one end of the communication link is usually a device carried by a user or has human presence nearby, which will produce changing propagation conditions due to body reflections and movements. In this experiment, we investigate whether human movements in the vicinity generate sufficient temporal variations between stationary devices.



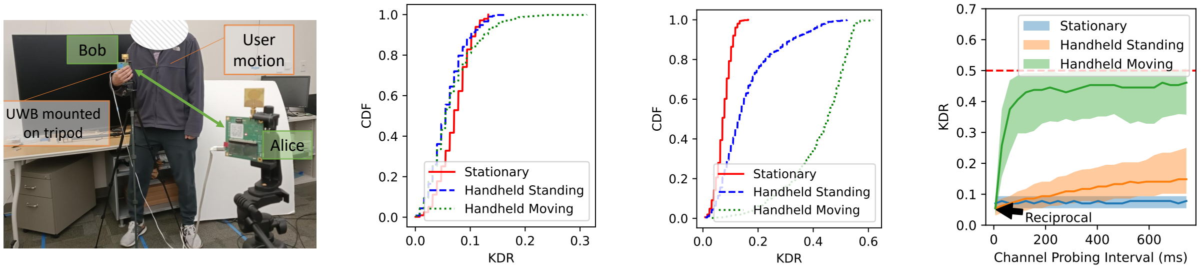
$a' = 0.3$ $a' = 0.8$ $a' = 1.5$

Fig. 12. Randomness test result of the generated keys when an injected path is present (with variable amplitude a' and delay τ'). The height of the bar plots represents the number of passed randomness tests (p-value > 0.05), out of the 9 total tests. The full results of the p-values for each test can be found in Section A.

The experimental setup is shown in Figure 13(a). Two UWB transceivers, acting as Alice and Bob, are placed on tripods and continuously perform key generation messaging. Three types of human engagements are investigated: (1) stationary, where a person stands next to Bob with no movements; (2) handheld standing, where a person holds one UWB in hand but makes no movements; (3) handheld moving, where a person holds one UWB in hand and makes small movements, such as hand gestures. Figure 13(b)–(c) show the KDR distribution of the positive samples and negative samples (generated at ∼ 100 ms apart). KDR is low (∼ 6%) for positive pairs (messages exchanged almost simultaneously one following the other) regardless of movement type. And as expected, the KDR for the negative pairs (messages exchanged with significant time between messages) is very low when there is no movement, indicating low temporal variations in the generated key. With user movements, however, KDR significantly increases indicating higher temporal variations. Notably, handheld standing also generates noticeably larger KDR than stationary case, which likely attributes to the minute movement of the user such as breathing and micro gestures. We further investigate how KDR distribution changes for keys generated at different intervals. Figure 13(d) shows that the $25th - 75th$ percentile range of the KDR represented by the shaded areas. KDR remains low for stationary channel regardless of the channel probing interval, while with human movements the KDR quickly approaches 50%. Therefore, even if the devices themselves, Alice and Bob, remain static and even when the surrounding environment does not change, nearby user engagements (both intentional and unintentional ones) can still produce significant variations. Therefore, memorization attacks become less likely with user movements, something that can be automatically encouraged by the application's semantics.

4.7 Model Complexity, Memory, and Latency

In this section, we investigate the effect of model complexity and quantization on the key generation performance. We trained three variations of CNN architecture with different complexity, with the "full" model containing 4 convolutional blocks and 4 dense layers, the "small" model containing 3 convolutional blocks and 2 dense layers, and the "tiny" model containing 2 convolutional blocks and 2 dense layers. The evaluations of *UWBKey*-Full in the previous sections uses the same architecture as the "full" model here. The architecture details are shown in Figure 14(a). Then, we perform quantization-aware training (QAT) (Figure 14(b)) to compress the model



(a) Illustration of experimental setup in investigating the influence of user body motions

(b) The CDF of KDR for reciprocally generated keys

(c) The CDF of KDR for keys generated at ~ 0.5 s interval.

(d) KDR over varying channel measurement interval for different body movements. Shaded areas represent the $25 - 75th$ percentile range. The lines in between represent the median.

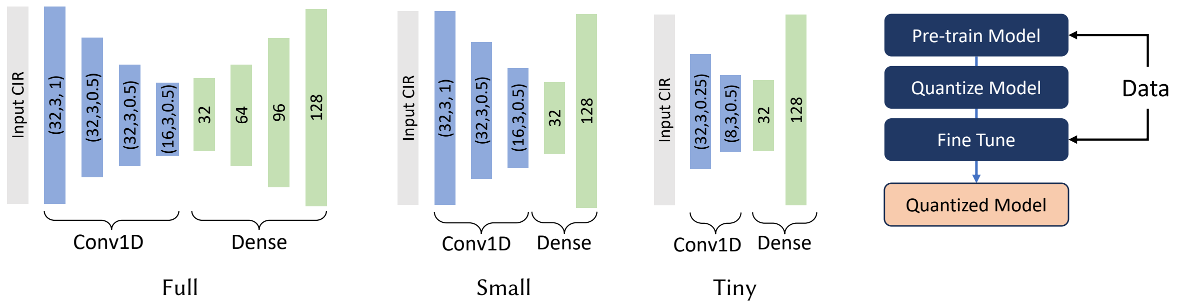
Fig. 13. Investigating temporal variations of the generated keys between two UWB devices with human presence nearby.

for deployment on resource-constrained IoT devices. We utilize Tensorflow-Lite Micro to quantize the 32-bit float neural network weights to 8-bit integers. The quantized model is ported to C/C++ to be deployed on a microcontroller.

After quantization, the tiny, small, and full models have a memory footprint of $17.2k$ *Bytes,* $31.7k$ *Bytes, and* $51.0k$ *Bytes* respectively, which are small enough to fit on a wide range of low-cost microcontrollers. We used the Arduino Nano 33 BLE Sense Rev2 board [4], which is based on the nRF52840 chipset (with 256kB of RAM), to run the quantized models and measure the latency (from the moment channel measurement becomes available to secret key being generated without key reconciliation). The average latencies are 12 ms for the tiny model, 40 ms for small and 50 ms for full, which would enable seamless operations for the purpose of most consumer applications. We again examine the KDR distribution and perform randomness testing on the binary keys generated by the quantized models (see Figure 15(c)–(d)). After quantization, the *UWBKey*-Full model generates comparable KDR distribution and randomness as the full-precision model. Out of all three quantized models, the tiny model shows noticeably poorer KDR distribution and randomness result. On the other hand, the small model shows low KDR for reciprocal keys and good randomness test results with 8 out of 9 tests passing, similar to the performance of the full quantized model. This shows that it is not necessary to use a very large neural network architecture or high-precision model weights for the contrastive learning-based key generation network, which enables easy deployment on resource-constrained IoT devices. Note that other more advanced model compression methods, such as neural architecture search and pruning, could be applied to further reduce the model size without sacrificing performance. It is left for future work.

4.8 Key Reconciliation

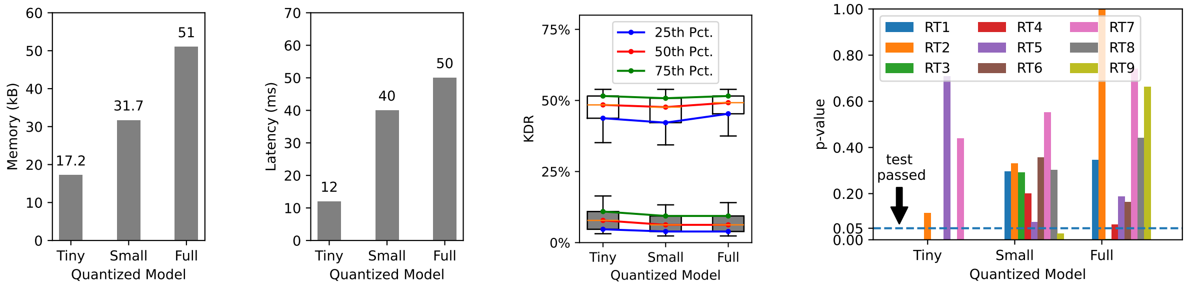
Following key generation, there may still be a small number of bits that are not identical between Alice and Bob. Therefore, a key reconciliation exchange is needed to correct the mismatched bits. In our experiment, we demonstrate the widely-adopted BCH and Reed-Solomon error correction coding schemes. Alice first encodes its generated key, produces the error correction parity bits, and sends the parity bits to Bob in an open channel. Then, Bob appends the parity bits to its generated key and performs error correction. The number of parity bits used determines the error correction ability, communication overhead, and information leakage due to openly



(a) Key Generation Network Architecture. Conv1D layers are denoted by (# of filters, filter size, MaxPooling ratio). Dense layers are denoted by the output dimension.

(b) Quantization-aware Training

Fig. 14. Reducing the model complexity and applying quantization-aware quantization for on-device deployment.



(a) Memory of the quantized models in kBytes.

(b) Latency of the quantized KGN tested on Arduino Nano 33 BLE Sense microcontroller.

(c) KDR distribution of the quantized models with data from unseen environments.

(d) Randomness test of the quantized models with data from unseen environments. Number of passed tests: 3 (Tiny), 8 (Small), 8 (Full).

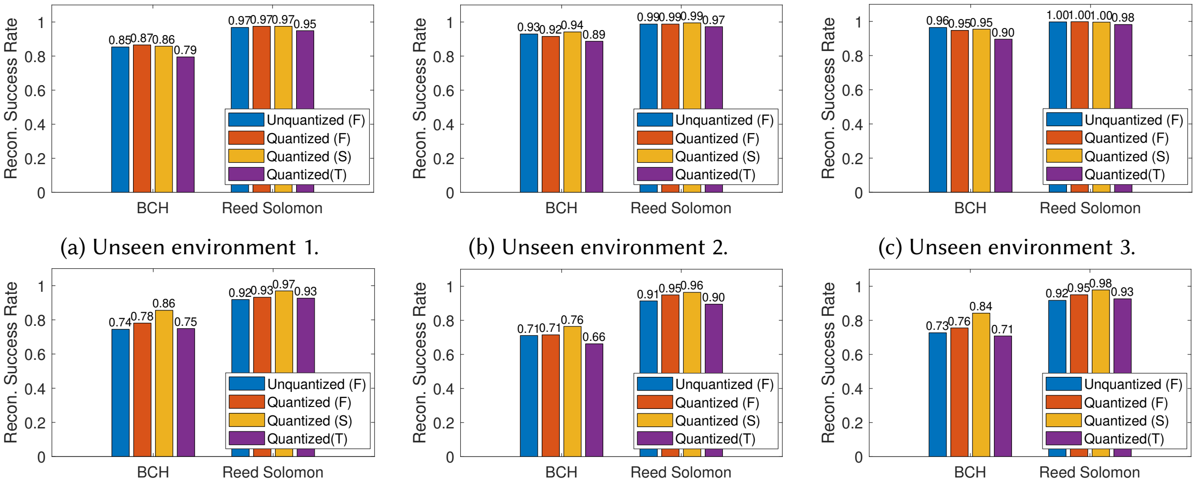
Fig. 15. The effect of model size and quantization on key generation performance.

communicated parity bits. Given the key length is 128 bits, we choose (255, 131) BCH code (codeword size of 255 bits, with 131 information bits and 124 parity bits), and (57,23,17) Reed-Solomon code (6 bits/symbol, 57 total symbols with 23 information symbols and can correct up to 17-symbol errors). The key reconciliation success rate is shown in Figure 16. It can be seen that Reed-Solomon code is able to achieve higher than 90% reconciliation success, and is more consistent than BCH code across all scenes. It should be noted that there is a tradeoff between communication overhead and the error correction ability: the communication overhead due to the parity bits is 124 bits for the chosen BCH code and 204 bits for the chosen Reed-Solomon code.

Achieving highly efficient key reconciliation is an area of active research. Explorations of other key reconciliation methods in this direction would be helpful. We plan to investigate this further for future work.

4.9 Adversary Analysis

4.9.1 Replay Attack. In a replay attack, the adversary would record an encrypted message without decrypting to replay at a later time. In *UWBKey*, replay attacks can be prevented by periodically changing the session key.



(a) Unseen environment 1.

(b) Unseen environment 2.

(c) Unseen environment 3.

(d) Unseen environment 4.

(e) Unseen environment 5.

(f) Unseen environment 6.

Fig. 16. Key reconciliation success rate of BCH code and Reed-Solomon code for the three test environments.

Because the wireless channel also decorrelates temporally, especially with user motion, the session keys generated at different times would be different.

4.9.2 Passive Attack. In a passive attack, the adversary tries to obtain the secret key by eavesdropping on Alice and Bob. In *UWBKey*, passive attack is prevented by several factors: (1) The wireless channel decorrelates spatially due to different multipaths experiences at different locations. (2) The *UWBKey* key generation function produces keys that resemble random numbers. Therefore, an adversary located at a different location from Alice or Bob could obtain little to no information about the secret keys by observing the channel.

4.9.3 Stalking Attack. As we have shown in Figure 10, a stalking eavesdropper with a separation of 0.85 m from the legitimate user already experiences KDR centered around 0.5. The likely explanation is that with sufficient spatial separation, there is enough incoherence between the channel observed by the stalking eavesdropper and by the legitimate user. Therefore, stalking attack can be avoided if there is no adversary within approximately arm length, which is easy for most user engaged applications.

4.9.4 Memorization Attack. This attack would succeed if the propagation conditions remained the same over time. In Figure 13, we show that if there is human motion near the mobile device, the changing multipaths due to body reflections are enough to produce decorrelated keys at different times. Therefore, in practice, the UWB device can prompt user engagement when it detects non-varrying channels to avoid the memorization attack.

4.9.5 Channel Injection Attack. Figure 11 shows how an injected path with different delay affects the generated key. In GBQ for example, an introduced path with a certain path delay has a high probability of affecting the bits at specific indices in the generated keys. In contrast, in *UWBKey*, an introduced path with any path delay as roughly equal 50% chance of affecting any bit index in the generated keys. In addition, Figure 12 shows that the keys generated by *UWBKey*-Full still pass the pseudo-random number tests even under a strong reflected path. This means that from the adversary's perspective, placing a reflector anywhere does not give any additional information about the keys generated by Alice and Bob.

4.10 Other Vulnerabilities

In addition to these adversary scenarios, the security vulnerabilities related to machine learning training and inference also need to be analyzed.

4.10.1 Model Inversion. In a model inversion attack [63, 68], the adversary may attempt to breach privacy by reverse engineering the training data. In a nutshell, a trained model may "memorize" certain training data. Therefore, by repeatedly observing the model output with known input data, the adversary can infer about what training data was used to build the model. In some contexts, this poses a serious security threat when the training data contains private information, such as email, fingerprints, facial images, etc. However, for CIR-based key generation task, the privacy concern regarding the training data is minimal. The training data is composed of UWB CIR, which do not carry sensitive or private information. It is true that existing wireless sensing literature has shown UWB CIR can be used to estimate bio-signals such as heart rate [59], detect materials [20, 66], sense radio spectrum [43], and detect human movements [12]. However, these techniques usually have other requirements (e.g. synchronization, radar mode, multiple sensors, carefully crafted setups), or need additional information (e.g. CIR update rate, motion trajectory, windows of consecutive measurements). Given our data collection setup and random sampling policy, even if the adversary can recover the training data, it would be extremely difficult to extract any information about the data collection scenes or the people within.

4.10.2 Data Poisoning/Backdoor Attack. In a data poisoning attack [52, 56], the adversary may manipulate the training data for the purpose of degrading model performance or leaving backdoor for future exploitations. For example, the adversary may embed certain malicious triggers or markers in the training data such that the trained learning algorithm produce false outputs when the markers appear in operation. However, for this type of attack to work, the adversary needs access to the training data. This may be true for many large-scale models where the data is scraped from the internet. In our work, we have full control of the dataset since we collect, clean and store all the data and make sure it is secure, so this is less of a concern in the scope of this paper. For larger studies or wide adoption of this method, we envision the data collection and storage will be done by a regulatory body or a standards group that has full control of the training dataset.

4.10.3 Side-channel Leakage. In a side-channel attack [18], the adversary may exploit indirect signals of a machine learning system to infer critical information, such as training data, model architecture and model weights. Beside model inversion, the adversary may employ membership/non-membership inference attack to speculate which data sample was or was not used during training. For our work, since our training data is simply the CIRs collected in various environments that do not carry sensitive or private information, it is less of a concern in the scope of this paper. The adversary may also use power consumption [61], electromagnetic emissions [7] and critical timing information [23] to extract the model architecture and weights. However, for our proposed system, the model itself is assumed to be public knowledge. In fact, as we have demonstrated, even if Eve has the full knowledge of the model, it is still near impossible for it to break the security of our system. Therefore, this type of attack is also irrelevant.

4.11 Limitations

4.11.1 Static Channel. When there is no movement in the environment, or when there are very limited number of multipaths (such as in an open outdoor space), the wireless channel could remain static over an extended time. In this case, generating keys from CIR is no longer secure as the temporal and spatial variation of CIR does not hold any more. This is a limitation for all channel-based key generation methods. To mitigate this issue, we propose a human-in-the-loop approach. In Figure 13, we saw that even if the UWB nodes remain static, a person moving nearby could generate sufficient variation in the generated keys. The system can detect a static channel and remind the user to move around.

We survey the literature and propose a few other remedies which could be explored in future work: (1) Placing a reconfigurable intelligent surface (RIS) [39, 42, 65]: RIS contains an array of reconfigurable radio reflective elements, and by randomly tuning the elements' reflective coefficients, the UWB receiver would observe different multipaths. Therefore, even if the UWB nodes remain static, the CIRs would still change constantly due to the RIS. (2) Utilizing beamforming from antenna arrays [31, 46]: If the transceivers are equipped with multi-antenna arrays, the transmitter could steer the beam towards different directions. As a result, the receiver would observe different propagation conditions even if both the transmitter and the receiver remain static [46]. The authors in [31] propose creating a virtual channel with opportunistic beamforming. (3) Cooperative jamming [22]: in cooperative jamming, when Alice transmits the channel sounding packet to Bob, another cooperative node can transmit a jamming packet concurrently to cause a random interference on the CIR. This random interference can increase the key entropy and make Eve's channel measurement more dissimilar at the same time. We leave the exploration of these methods for future work.

5 RELATED WORK

UWB is a wireless communication technology that operates over bandwidths of 500 MHz and higher. Compared to other radio standards, UWB's large bandwidth allows it to distinguish multipath at a high resolution, enabling it to provide robust and precise indoor positioning [13]. Its decimeter-level localization accuracy and low power consumption makes UWB suitable for a wide range of applications, such as AR/VR interface [5], robot navigation [36], social interaction tracking [8], and pet monitoring [2], to improve the usability and context-awareness in these applications. More recently, there are growing interests in adopting UWB for security-sensitive applications, such as the digital key [17] and two-factor authentication [11]. In the automotive domain, the Car Connectivity Consortium (CCC) has published the standardization of Digital Key to enable seamless and secure keyless entry to vehicles [17], which utilizes UWB ranging for evidence of physical proximity as an additional factor of authentication to prevent relay attacks. While these solutions bring new opportunities, they also spotlight the importance of ranging and communication security in UWB.

The IEEE 802.15.4z standard lays the groundwork for secured ranging and communication using the pseudo-random Scrambled Time Sequence (STS) field and AES-128 encryption [16], in which both need a 128-bit symmetric sequence to serve as the encryption key or an initial seed to a pseudo-random number generator. Interestingly, how to reach the agreement of this 128-bit sequence between devices is not specified in the standard and is up to the discretion of the product designer. The CCC Digital Key [26] proposes using an out-of-band BLE connection to establish this key. While it may be appropriate for automotive settings since BLE was already part of the integral solution, a standalone UWB system should not rely on a different technology. Key exchange protocols such as Diffie-Hellman exchange [21] can be employed, but it has been shown to be vulnerable to adversaries with large computing power [1] or quantum computing facilities [54]. Secret keys with characteristics of random numbers are resistant to such attacks as brute force remains the only means of breaking it, and not intractable reverse computational problems [49]. Spoofing of reported ranging timings at the application level has been addressed in a recent work UnSpoof [14], however it does not address the underlying encryption of UWB packets.

Key agreement through physical layer key generation provides an interesting alternative. Of course, we are not the first ones to realize this potential. Previously, researcher have proposed to generate symmetric keys from the wireless channel owing to its reciprocity [24, 40, 64]. However, compared to other narrow-band wireless technologies, UWB is a good candidate for physical layer key generation since its wider bandwidth results in CIR measurements containing rich information about the channel. This higher granularity information can be used for generating binary keys [10, 29, 30, 45, 47, 53, 58]. In the literature, there have been multiple works performing theoretical analysis of UWB-based key generation [30, 53, 58], while a few implementations are based on bulky signal generator-based UWB hardware [10, 29, 45, 47], and sometimes using bandwidths significantly wider than those allowed by the FCC for commercial use. While these foundational works are important and provide us a stepping stone, our focus in *UWBKey* is on developing key generation algorithms based on the capabilities of COTS UWB devices available today. We aim to demonstrate a practical key generation solution for COTS UWB devices, including in practical situations outside of the lab settings. The main challenge lies in generating symmetric keys with minimal bit error rate despite measurement errors while at the same time maintaining high randomness. The authors of [57] proposed using the Siamese Network to learn a feature extraction model that solicits reciprocal channel characteristics for key generation, and evaluates on COTS UWB devices in the real world. In this work, we propose a novel key generation method that combines multi-sample contrastive learning with entropy-based augmentation and data augmentation, which generates 128-bit keys with high randomness and robustness. In addition, to the best of our knowledge, *UWBKey* is the first UWB key generation solution that demonstrates on-device deployment on COTS resource-constrained microcontrollers.

6 DISCUSSION AND CONCLUDING REMARKS

Truly ubiquitous and stand-alone use of ultra-wideband (UWB) technology is inhibited by the need for establishing secure communication keys which today can be only performed out-of-band (OOB). This approach is getting entrenched with both Apple and Google offering only OOB means to initiate UWB pairing and further communication [3, 28]. Several applications, including those shown in Figure 1 need the ability to communicate with proximal devices without necessarily going through a complex out of band protocol to establish secure communication. Even indoor localization, a primary use-case of UWB, is being rendered infeasible with today's OOB pairing requirements. This paper attempts to break UWB free from OOB pairing requirements by showing a practical and computationally inexpensive method for secure key generation which has the potential to become a foundational primitive in future UWB offerings. Of course, advertising supported UWB parameters, such as channels and data rates, is another reason why OOB communication is needed. Parameter negotiations still needs to be performed, but we expect that the industry will converge on a beaconing and scanning mechanism similar to Wi-Fi Beacons [55] to overcome that hurdle.

In conclusion, we propose *UWBKey*, a key generation solution that enables ranging and communication security in UWB while eliminating the need for an out-of-band key exchange. Evaluated in various environments and conditions, it achieves a low median bit error rate of $\sim 6\%$ for a pair of transceivers, which can be easily corrected with forward error correction codes, and high randomness that makes guessing the correct key extremely difficult for the adversary. *UWBKey*'s model can be compressed and deployed onto resource-constrained microcontrollers while still providing acceptable performance and low latency of a few milliseconds. We believe *UWBKey* will improve many existing applications including vehicle keyless entry, location-based interaction, access control, etc., in terms of their usability and security, and enable new applications in the future. We plan to open-source the data we have collected and open-source the entire key generation pipeline for the benefit of the research community.

ACKNOWLEDGMENTS

This work has been partially supported by the NSF under Grant No. 2145278.

REFERENCES

[1] David Adrian, Karthikeyan Bhargavan, Zakir Durumeric, Pierrick Gaudry, Matthew Green, J. Alex Halderman, Nadia Heninger, Drew Springall, Emmanuel Thomé, Luke Valenta, Benjamin VanderSloot, Eric Wustrow, Santiago Zanella-Béguelin, and Paul Zimmermann. 2015. Imperfect Forward Secrecy: How Diffie-Hellman Fails in Practice. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security* (Denver, Colorado, USA) *(CCS '15)*. Association for Computing Machinery, New York, NY, USA, 5–17. https://doi.org/10.1145/2810103.2813707

[2] Neeraj Alavala, Haige Chen, Rishabh Singhal, and Ashutosh Dhekne. 2022. PetTrack: Tracking Pet Location and Activity Indoors. In *Proceedings of the 2022 Workshop on Body-centric Computing Systems*. 1–6.

[3] Apple. 2025. Nearby Interaction: Locate and interact with nearby devices using identifiers, distance, and direction. https://developer.apple.com/documentation/nearbyinteraction/

[4] Arduino. 2025. Arduino Nano 33 BLE Sense Rev2. https://docs.arduino.cc/resources/datasheets/ABX00069-datasheet.pdf.

[5] Aditya Arun, Shunsuke Saruwatari, Sureel Shah, and Dinesh Bharadia. 2023. XRLoc: Accurate UWB Localization for XR Systems. arXiv:2307.12512 [cs.HC]

[6] Lawrence Bassham, Andrew Rukhin, Juan Soto, James Nechvatal, Miles Smid, Stefan Leigh, M Levenson, M Vangel, Nathanael Heckert, and D Banks. 2010. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=906762

[7] Lejla Batina, Shivam Bhasin, Dirmanto Jap, and Stjepan Picek. 2019. CSI NN: Reverse Engineering of Neural Network Architectures Through Electromagnetic Side Channel. In *28th USENIX Security Symposium (USENIX Security 19)*. USENIX Association, Santa Clara, CA, 515–532. https://www.usenix.org/conference/usenixsecurity19/presentation/batina

[8] Andreas Biri, Neal Jackson, Lothar Thiele, Pat Pannuto, and Prabal Dutta. 2020. SociTrack: Infrastructure-Free Interaction Tracking through Mobile Sensor Networks. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking* (London, United Kingdom) *(MobiCom '20)*. Association for Computing Machinery, New York, NY, USA, Article 33, 14 pages. https://doi.org/10.1145/3372224.3419190

[9] Dutliff Boshoff, Morgana Mo Zhou, Raphael E. Nkrow, Bruno Silva, and Gerhard P. Hancke. 2025. UWB Physical Layer Key Sharing Using the Frequency Domain CIR Magnitude. *IEEE Transactions on Industrial Informatics* 21, 2 (2025), 2000–2009. https://doi.org/10.1109/TII.2025.3526320

[10] M. Bulenok, I. Tunaru, L. Biard, B. Denis, and B. Uguen. 2016. Experimental channel-based secret key generation with integrated ultra wideband devices. *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC*. https://doi.org/10.1109/PIMRC.2016.7794705

[11] Yifeng Cao, Ashutosh Dhekne, and Mostafa Ammar. 2024. UWB-Auth: A UWB-based Two Factor Authentication Platform *(WiSec '24)*. Association for Computing Machinery, New York, NY, USA, 185–195. https://doi.org/10.1145/3643833.3656113

[12] Shrenik Changede and Ashutosh Dhekne. 2022. IntruSense: an enhanced physical security system using UWB. In *Proceedings of the 23rd Annual International Workshop on Mobile Computing Systems and Applications*. 41–47.

[13] Haige Chen and Ashutosh Dhekne. 2022. PnPLoc: UWB Based Plug Play Indoor Localization. In *2022 IEEE 12th International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. 1–8. https://doi.org/10.1109/IPIN54987.2022.9918119

[14] Haige Chen and Ashutosh Dhekne. 2023. UnSpoof: Distance Spoofing-Evident Localization using UWB. In *2023 IEEE 13th International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. 1–6. https://doi.org/10.1109/IPIN57070.2023.10332533

[15] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A Simple Framework for Contrastive Learning of Visual Representations. *arXiv preprint arXiv:2002.05709* (2020).

[16] LAN/MAN Standards Committee. 2020. *IEEE Standard for Low-Rate Wireless Networks. Amendment 1: Enhanced Ultra Wideband (UWB) Physical Layers (PHYs) and Associated Ranging Techniques (IEEE Std 802.15.4z)*. Vol. 2020. 1–174 pages. https://standards.ieee.org/standard/index.html

[17] Car Connectivity Consortium. 2022. WHITEPAPER: CCC Digital Key-The Future of Vehicle Access. https://carconnectivity.org/wp-content/uploads/2022/11/CCC_Digital_Key_Whitepaper_Approved.pdf

[18] Edoardo Debenedetti, Giorgio Severi, Nicholas Carlini, Christopher A. Choquette-Choo, Matthew Jagielski, Milad Nasr, Eric Wallace, and Florian Tramèr. 2024. Privacy Side Channels in Machine Learning Systems. In *33rd USENIX Security Symposium (USENIX Security 24)*. USENIX Association, Philadelphia, PA, 6861–6848. https://www.usenix.org/conference/usenixsecurity24/presentation/debenedetti

[19] Eshagh Dehmollaian, Bernhard Etzlinger, and Andreas Springer. 2024. A Lightweight CIR-Based Physical Layer Key Generation Scheme for UWB. In *2024 IEEE Wireless Communications and Networking Conference (WCNC)*. 1–6. https://doi.org/10.1109/WCNC57260.2024.10570817

[20] Ashutosh Dhekne, Mahanth Gowda, Yixuan Zhao, Haitham Hassanieh, and Romit Roy Choudhury. 2018. Liquid: A wireless liquid identifier. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys)*. 442–454.

[21] Whitfield Diffie and Martin E Hellman. 2022. New directions in cryptography. In *Democratizing Cryptography: The Work of Whitfield Diffie and Martin Hellman*. 365–390.

[22] Lun Dong, Zhu Han, Athina P. Petropulu, and H. Vincent Poor. 2009. Cooperative jamming for wireless physical layer security. In *2009 IEEE/SP 15th Workshop on Statistical Signal Processing*. 417–420. https://doi.org/10.1109/SSP.2009.5278549

[23] Vasisht Duddu, Debasis Samanta, D Vijay Rao, and Valentina E. Balas. 2019. Stealing Neural Networks via Timing Side Channels. arXiv:1812.11720 [cs.CR] https://arxiv.org/abs/1812.11720

[24] Youssef El, Hajj Shehadeh, Omar Alfandi, and Dieter Hogrefe. 2012. Towards Robust Key Extraction from Multipath Wireless Channels. Issue 4.

[25] Michelle Fernando, Dhammika Jayalath, Seyit Camtepe, and Ernest Foo. 2017. Reed Solomon Codes for the Reconciliation of Wireless PHY Layer Based Secret Keys. In *2017 IEEE 86th Vehicular Technology Conference (VTC-Fall)*. 1–6. https://doi.org/10.1109/VTCFall.2017.8288218

[26] Kerry Glover. 2022. Ultra-Wideband CCC Digital Key: Automobile Security Convenience. https://www.symmetryelectronics.com/getmedia/725c2852-1c7d-4d9b-b06f-ebf4196adf75/qorvo-uwb-automotive-ccc-digital-key-e-guide.pdf

[27] Google. 2025. Ultra-wideband (UWB) communication. https://developer.android.com/develop/connectivity/uwb

[28] Google. 2025. Ultra-wideband (UWB) communication. https://developer.android.com/develop/connectivity/uwb

[29] Sana Tmar Ben Hamida, Jean Benoît Pierrot, and Claude Castelluccia. 2010. Empirical analysis of UWB channel characteristics for secret key generation in indoor environments. *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC*, 1984–1989. https://doi.org/10.1109/PIMRC.2010.5671596

[30] S. Tmar Ben Hamida, J. B. Pierrot, B. Denis, C. Castelluccia, and B. Uguen. 2012. On the security of UWB secret key generation methods against deterministic channel prediction attacks. *IEEE Vehicular Technology Conference*. https://doi.org/10.1109/VTCFall.2012.6399358

[31] Pengfei Huang and Xudong Wang. 2013. Fast secret key generation in static wireless networks: A virtual channel approach. In *2013 Proceedings IEEE INFOCOM*. 2292–2300. https://doi.org/10.1109/INFCOM.2013.6567033

[32] Apple Inc. 2025. Nearby Interaction: Locate and interact with nearby devices using identifiers, distance, and direction. https://developer.apple.com/documentation/NearbyInteraction

[33] Apple Inc. 2025. Nearby interaction with UWB. https://developer.apple.com/nearby-interaction/

[34] 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 *Proceedings of the 15th Annual International Conference on Mobile Computing and Networking* (Beijing, China) *(MobiCom '09)*. Association for Computing Machinery, New York, NY, USA, 321–332. https://doi.org/10.1145/1614320.1614356

[35] Phuc H. Le-Khac, Graham Healy, and Alan F. Smeaton. 2020. Contrastive Representation Learning: A Framework and Review. *IEEE Access* 8 (2020), 193907–193934. https://doi.org/10.1109/ACCESS.2020.3031549

[36] Anton Ledergerber, Michael Hamer, and Raffaello D'Andrea. 2015. A robot self-localization system using one-way ultra-wideband communication. *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* 2015-Decem, 3131–3137. https://doi.org/10.1109/IROS.2015.7353810

[37] Kyuin Lee, Yucheng Yang, Omkar Prabhune, Aishwarya Lekshmi Chithra, Jack West, Kassem Fawaz, Neil Klingensmith, Suman Banerjee, and Younghyun Kim. 2022. AEROKEY: Using Ambient Electromagnetic Radiation for Secure and Usable Wireless Device Authentication. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 6, 1, Article 20 (March 2022), 29 pages. https://doi.org/10.1145/3517254

[38] Patrick Leu, Giovanni Camurati, Alexander Heinrich, Marc Roeschlin, Claudio Anliker, Matthias Hollick, Srdjan Capkun, and Jiska Classen. 2022. Ghost Peak: Practical Distance Reduction Attacks Against HRP UWB Ranging. In *31st USENIX Security Symposium (USENIX Security 22)*. USENIX Association, Boston, MA, 1343–1359. https://www.usenix.org/conference/usenixsecurity22/presentation/leu

[39] Guyue Li, Lei Hu, Paul Staat, Harald Elders-Boll, Christian Zenger, Christof Paar, and Aiqun Hu. 2022. Reconfigurable Intelligent Surface for Physical Layer Key Generation: Constructive or Destructive? *IEEE Wireless Communications* 29, 4 (2022), 146–153. https://doi.org/10.1109/MWC.007.2100545

[40] Hongbo Liu, Yang Wang, Jie Yang, and Yingying Chen. 2013. Fast and practical secret key extraction by exploiting channel response. *Proceedings - IEEE INFOCOM*, 3048–3056. https://doi.org/10.1109/INFCOM.2013.6567117

[41] Xin Liu, Zhongdao Wang, Ya-Li Li, and Shengjin Wang. 2022. Self-Supervised Learning via Maximum Entropy Coding. In *Advances in Neural Information Processing Systems*, Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (Eds.). https://openreview.net/forum?id=nJt27NQffr

[42] Xinjin Lu, Jing Lei, Yuxin Shi, and Wei Li. 2021. Intelligent Reflecting Surface Assisted Secret Key Generation. *IEEE Signal Processing Letters* 28 (2021), 1036–1040. https://doi.org/10.1109/LSP.2021.3061301

[43] Zhicheng Luo, Qianyi Huang, Rui Wang, Hao Chen, Xiaofeng Tao, Guihai Chen, and Qian Zhang. 2023. WISE: Low-Cost Wide Band Spectrum Sensing Using UWB. In *Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems* (Boston, Massachusetts) *(SenSys '22)*. Association for Computing Machinery, New York, NY, USA, 651–666. https://doi.org/10.1145/3560905.3568541

[44] Yi Ma, Harm Derksen, Wei Hong, and John Wright. 2007. Segmentation of Multivariate Mixed Data via Lossy Data Coding and Compression. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29, 9 (2007), 1546–1562. https://doi.org/10.1109/TPAMI.2007.1085

[45] Masoud Ghoreishi Madiseh, Shuai He, Michael L. McGuire, Stephen W. Neville, and Xiaodai Dong. 2009. Verification of secret key generation from UWB channel observations. *IEEE International Conference on Communications*. https://doi.org/10.1109/ICC.2009.5199564

[46] Masoud Ghoreishi Madiseh, Stephen W. Neville, and Michael L. McGuire. 2012. Applying Beamforming to Address Temporal Correlation in Wireless Channel Characterization-Based Secret Key Generation. *IEEE Transactions on Information Forensics and Security* 7, 4 (2012), 1278–1287. https://doi.org/10.1109/TIFS.2012.2195176

[47] Francesco Marino, Enrico Paolini, and Marco Chiani. 2014. Secret key extraction from a UWB channel: Analysis in a real environment. *Proceedings - IEEE International Conference on Ultra-Wideband*, 80–85. https://doi.org/10.1109/ICUWB.2014.6958955

[48] Samsung Newsroom. 2023. Samsung Announces Ultra-Wideband Chipset With Centimeter-Level Accuracy for Mobile and Automotive Devices. https://news.samsung.com/global/samsung-announces-ultra-wideband-chipset-with-centimeter-level-accuracy-for-mobile-and-automotive-devices

[49] Ray Perlner and David Cooper. 2009. Quantum Resistant Public Key Cryptography: A Survey. IDtrust 2009, Gaithersburg, MD. https://doi.org/10.1145/1527017.1527028

[50] Andrew Romero. 2024. The Moto Tag is the first UWB tracker to join Google's Find My Device network. https://9to5google.com/2024/06/25/moto-tag-ultra-wideband-tracker-google-find-my-device-network/

[51] NXP Semiconductors. 2021. NXP Trimension™ Ultra-Wideband Technology Powers Xiaomi MIX4 Smartphone to Deliver New "Point to Connect" Smart Home Solution. https://www.nxp.com/company/about-nxp/newsroom/NW-NXP-TRIMENSION-ULTRA-WIDEBAND-TECHNOLOGY-POWERS

[52] Ali Shafahi, W. Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. 2018. Poison Frogs! Targeted Clean-Label Poisoning Attacks on Neural Networks. In *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.), Vol. 31. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2018/file/22722a343513ed45f14905eb07621686-Paper.pdf

[53] Ghasem Naddafzadeh Shirazi and Lutz Lampe. 2012. A compressive sensing approach for secret key agreement based on UWB channel reciprocity. *Proceedings - IEEE International Conference on Ultra-Wideband*, 135–139. https://doi.org/10.1109/ICUWB.2012.6340497

[54] P.W. Shor. 1994. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*. 124–134. https://doi.org/10.1109/SFCS.1994.365700

[55] Cisco TCC2. 2020. The significance of beacon frames and how to configure the beacon interval on Access Points. https://community.cisco.com/t5/wireless-mobility-knowledge-base/the-significance-of-beacon-frames-and-how-to-configure-the/ta-p/3132525

[56] Florian Tramèr, Reza Shokri, Ayrton San Joaquin, Hoang Le, Matthew Jagielski, Sanghyun Hong, and Nicholas Carlini. 2022. Truth Serum: Poisoning Machine Learning Models to Reveal Their Secrets. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security* (Los Angeles, CA, USA) *(CCS '22)*. Association for Computing Machinery, New York, NY, USA, 2779–2792. https://doi.org/10.1145/3548606.3560554

[57] Paul Walther and Thorsten Strufe. 2020. Blind Twins: Siamese Networks for Non-Interactive Information Reconciliation. In *2020 IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications*. 1–7. https://doi.org/10.1109/PIMRC48278.2020.9217278

[58] Robert Wilson, David Tse, and Robert A. Scholtz. 2007. Channel identification: Secret sharing using reciprocity in ultrawideband channels. *IEEE Transactions on Information Forensics and Security* 2 (9 2007), 364–375. Issue 3. https://doi.org/10.1109/TIFS.2007.902666

[59] Shuqiong Wu, Takuya Sakamoto, Kentaro Oishi, Toru Sato, Kenichi Inoue, Takeshi Fukuda, Kenji Mizutani, and Hiroyuki Sakai. 2019. Person-Specific Heart Rate Estimation With Ultra-Wideband Radar Using Convolutional Neural Networks. *IEEE Access* 7 (2019), 168484–168494. https://doi.org/10.1109/ACCESS.2019.2954294

[60] Zhenqian Wu, Youming Li, Xiaolong Zhang, Xiangpei Meng, Xinrong Lv, and Yonghong Wu. 2024. Multiple Anchors and RIS-Aided Localization Method in Complex NLOS Environments. *IEEE Internet of Things Journal* 11, 22 (2024), 36922–36932. https://doi.org/10.1109/JIOT.2024.3433948

[61] Yun Xiang, Zhuangzhi Chen, Zuohui Chen, Zebin Fang, Haiyang Hao, Jinyin Chen, Yi Liu, Zhefu Wu, Qi Xuan, and Xiaoniu Yang. 2020. Open DNN Box by Power Side-Channel Attack. *IEEE Transactions on Circuits and Systems II: Express Briefs* 67, 11 (2020), 2717–2721. https://doi.org/10.1109/TCSII.2020.2973007

[62] Weitao Xu, Sanjay Jha, and Wen Hu. 2019. LoRa-Key: Secure Key Generation System for LoRa-Based Network. *IEEE Internet of Things Journal* 6, 4 (2019), 6404–6416. https://doi.org/10.1109/JIOT.2018.2888553

[63] Wencheng Yang, Song Wang, Di Wu, Taotao Cai, Yanming Zhu, Shicheng Wei, Yiying Zhang, Xu Yang, Zhaohui Tang, and Yan Li. 2025. Deep Learning Model Inversion Attacks and Defenses: A Comprehensive Survey. arXiv:2501.18934 [cs.CR] https://arxiv.org/abs/2501.18934

[64] Junqing Zhang, Roger Woods, Trung Q. Duong, Alan Marshall, Yuan Ding, Yi Huang, and Qian Xu. 2016. Experimental Study on Key Generation for Physical Layer Security in Wireless Communications. *IEEE Access* 4 (2016), 4464–4477. https://doi.org/10.1109/ACCESS.2016.2604618

[65] Ran Zhao, Qi Qin, Ningya Xu, Guoshun Nan, Qimei Cui, and Xiaofeng Tao. 2022. SemKey: Boosting Secret Key Generation for RIS-assisted Semantic Communication Systems. In *2022 IEEE 96th Vehicular Technology Conference (VTC2022-Fall)*. 1–5. https://doi.org/10.1109/VTC2022-Fall57202.2022.10013083

[66] Tianyue Zheng, Zhe Chen, Jun Luo, Lin Ke, Chaoyang Zhao, and Yaowen Yang. 2021. SiWa: see into walls via deep UWB radar. In *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking* (New Orleans, Louisiana) *(MobiCom '21)*. Association for Computing Machinery, New York, NY, USA, 323–336. https://doi.org/10.1145/3447993.3483258

[67] Yawen Zheng, Fan Dang, Zihao Yang, Jinyan Jiang, Xu Wang, Lin Wang, Kebin Liu, Xinlei Chen, and Yunhao Liu. 2024. BlueKey: Exploiting Bluetooth Low Energy for Enhanced Physical-Layer Key Generation. In *IEEE INFOCOM 2024 - IEEE Conference on Computer Communications*. 711–720. https://doi.org/10.1109/INFOCOM52122.2024.10621142

[68] Zhanke Zhou, Jianing Zhu, Fengfei Yu, Xuan Li, Xiong Peng, Tongliang Liu, and Bo Han. 2024. Model Inversion Attacks: A Survey of Approaches and Countermeasures. arXiv:2411.10023 [cs.LG] https://arxiv.org/abs/2411.10023

A RANDOMNESS TEST UNDER CHANNEL INJECTION ATTACK

Detailed tables of our perturbations of all CIRs measured in the unseen environment by adding an artificial path with the variable amplitude ($a' = 0.3, 0.8, 1.5$) and delay ($\tau' = 20, 30, 40, 50, 60ns$) are shown here. We examine the randomness of the generated keys using the NIST Test Suite tests listed in Table 1. Below, Table 2 shows the results for $a' = 0.3$, followed by Table 3 showing the results for $a' = 0.8$, and finally Table 4 showing the results for $a' = 1.5$.

Table 2. NIST Statistical Test for Random Number Testing results when an injected path with delay τ' and amplitude of $a' = 0.3$ is present. The p-value in the range of $[0, 1]$ is shown for each test. A p-value greater than 0.05 is bolded to indicate a passed test.

Type of Test	$\tau' = 20$ ns		$\tau' = 30$ ns		$\tau' = 40$ ns		$\tau' = 50$ ns		$\tau' = 60$ ns	
	F.	V.	F.	V.	F.	V.	F.	V.	F.	V.
RT1	0.0140	**0.1679**	0.0206	**0.5245**	**0.1329**	**0.2031**	0.0477	0.0439	**0.0714**	**0.1329**
RT2	**0.8255**	0.0026	**0.6356**	0.0029	**0.9404**	0.0033	**0.9881**	0.0023	**0.9357**	0.0026
RT3	**0.5127**	**0.2575**	**0.7411**	**0.1249**	**0.5731**	**0.1108**	**0.6579**	**0.0801**	**0.5404**	**0.2442**
RT4	**0.3000**	0.0002	**0.0657**	0.0059	**0.2769**	0.0008	**0.4929**	0.0001	**0.2938**	0.0016
RT5	**0.1743**	0.0342	**0.5622**	0.0297	**0.3202**	**0.0892**	**0.7412**	**0.0582**	**0.7379**	**0.0509**
RT6	**0.4559**	0.0000	**0.7343**	0.0071	**0.3490**	**0.1000**	**0.3398**	0.0206	**0.0593**	**0.1826**
RT7	**0.2938**	0.0405	**0.7035**	**0.4935**	0.0209	**0.6954**	**0.6711**	**0.7358**	**0.1492**	**0.3674**
RT8	0.0213	**0.1125**	0.0144	**0.2846**	**0.1633**	**0.1037**	**0.0823**	0.0254	**0.1102**	**0.0995**
RT9	0.0079	**0.1896**	0.0107	**0.6960**	**0.0756**	**0.2272**	0.0309	0.0402	**0.0518**	**0.1346**

Table 3. NIST Statistical Test for Random Number Testing results when an injected path with delay τ' and amplitude of $a' = 0.8$ is present. The p-value in the range of $[0, 1]$ is shown for each test. A p-value greater than 0.05 is bolded to indicate a passed test.

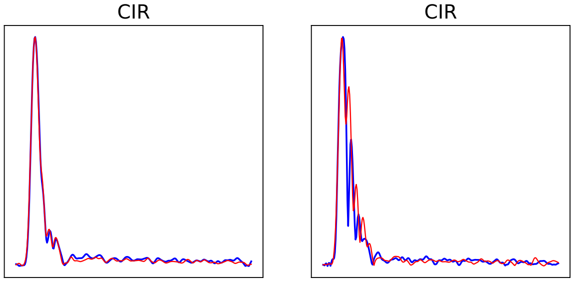
Type of Test	$\tau' = 20$ ns		$\tau' = 30$ ns		$\tau' = 40$ ns		$\tau' = 50$ ns		$\tau' = 60$ ns	
	F.	V.	F.	V.	F.	V.	F.	V.	F.	V.
RT1	0.0016	0.0000	**0.1077**	0.0000	**0.8597**	0.0000	0.0259	0.0000	**0.3864**	0.0000
RT2	**0.9665**	0.0000	**0.9948**	0.0007	**0.9994**	0.0000	**0.9823**	0.0000	**0.9888**	0.0002
RT3	**0.9717**	0.0000	**0.0652**	0.0000	**0.9861**	0.0000	0.0001	0.0000	**0.6379**	0.0000
RT4	**0.5634**	0.0000	**0.2490**	0.0000	0.0099	0.0000	0.0226	0.0000	**0.4737**	0.0000
RT5	0.0000	**0.8696**	**0.3320**	**0.5252**	**0.6681**	**0.1687**	0.0002	**0.4139**	**0.1687**	**0.2208**
RT6	**0.9030**	0.0000	**0.2512**	0.0000	0.0174	0.0000	**0.2593**	0.0000	**0.1219**	0.0000
RT7	**0.7883**	**0.2469**	**0.4645**	**0.6495**	**0.5853**	**0.1973**	**0.3062**	**0.5539**	0.0017	**0.6846**
RT8	0.0022	0.0000	**0.1484**	0.0000	**0.8920**	0.0000	0.0441	0.0000	**0.6384**	0.0000
RT9	0.0027	0.0000	**0.2115**	0.0000	**0.9842**	0.0000	**0.0507**	0.0000	**0.4380**	0.0000

Table 4. NIST Statistical Test for Random Number Testing results when an injected path with delay τ' and amplitude of $a' = 1.5$ is present. The p-value in the range of $[0, 1]$ is shown for each test. A p-value greater than 0.05 is bolded to indicate a passed test.

Type of Test	$\tau' = 20$ ns		$\tau' = 30$ ns		$\tau' = 40$ ns		$\tau' = 50$ ns		$\tau' = 60$ ns	
	F.	V.	F.	V.	F.	V.	F.	V.	F.	V.
RT1	0.0094	0.0000	**0.7909**	0.0000	0.0226	0.0000	**0.0518**	0.0000	**0.1522**	0.0000
RT2	**0.9653**	0.0000	**0.9996**	0.0006	**1.0000**	0.0000	**1.0000**	0.0000	**1.0000**	0.0000
RT3	**0.3362**	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000
RT4	**0.0815**	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0005	0.0000
RT5	**0.2121**	0.0006	0.0003	0.0000	0.0000	0.0000	0.0000	**0.3086**	**0.0560**	**0.1146**
RT6	**0.2532**	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0459	0.0000
RT7	0.0174	**0.3435**	**0.4959**	0.0020	**0.4645**	**0.9496**	**0.4551**	**0.4935**	**0.7986**	**0.4275**
RT8	0.0156	0.0000	**0.7868**	0.0000	0.0241	0.0000	**0.0507**	0.0000	**0.2193**	0.0000
RT9	0.0182	0.0000	**0.6795**	0.0000	0.0273	0.0000	**0.0678**	0.0000	**0.1896**	0.0000

B SAMPLE CIRS AND CORRESPONDING GENERATED KEYS

Figure 17(a) shows two example reciprocal CIRs which almost overlap completely. The resultant key will closely match. In contrast, when the CIRs are not reciprocal, say due to temporal or spatial variations, Figure 17(b) shows the CIR mismatch, which may not be immediately apparent. These subtle difference between the CIRs will still result in almost 50% bits to be different between the two CIRs' generated keys.



(a) Example Matched CIR (b) Example Mismatched CIR

Fig. 17. Example CIRs: Reciprocal CIRs match closely while mismatched CIRs show subtle differences.