Vehicle-Key: A Secret Key Establishment Scheme for LoRa-enabled IoV Communications

Huanqi Yang[1,2], Hongbo Liu[3], Chengwen Luo[4], Yuezhong Wu[5], Wei Li[6], Albert Y. Zomaya[6], Linqi Song[1,2] and Weitao Xu[1,2,*]

[1]City University of Hong Kong Shenzhen Research Institute, [2]City University of Hong Kong, [3]University of Electronic Science and Technology of China, [4]Shenzhen University, [5]University of New South Wales, [6]The University of Sydney

***Abstract*—Recent years have witnessed the remarkable growth of the Internet of Vehicles (IoV). Due to the high dynamics and ad-hoc nature of IoV communication, the lack of effective secret key establishment in IoV remains a security bottleneck. Physical layer key generation has emerged as a promising technology to establish a pair of cryptographic keys in a lightweight and information-theoretic secure way. However, prior works mainly focus on legacy communication technologies such as Wi-Fi, ZigBee, and 5G which can only achieve short range IoV communications. The emergence of Long-range (LoRa) communication technology that features long-range, low power, and extremely low data rate, brings new challenges for key generation in long range IoV scenarios. In this paper, we present Vehicle-Key, which is a secret key generation system to secure LoRa-enabled IoV communications. In Vehicle-Key, we design a novel deep learning model that can achieve channel prediction and quantization simultaneously. Additionally, we propose an autoencoder-based reconciliation method that improves the key agreement rate significantly. Extensive real-world experiments show that Vehicle-Key improves the key agreement rate by 15.10%–49.81% and key generation rate by 9–14× compared with the state-of-the-art. Security analysis demonstrates that Vehicle-Key is secure against several common attacks. Moreover, we implement Vehicle-Key on a Raspberry Pi and show that it can be executed in 3.4 ms.**

***Index Terms*—Internet of Vehicles, Physical layer key generation, LoRa**

I. Introduction

A. Background and Motivation

The rapid development of the Internet of Vehicles (IoV), which aims to connect any objects in the vehicle networks to improve the urban transport system, reduce accidents, and enhance the traffic monitoring system, has gained considerable attention due to its ubiquitous feature. As shown in Fig. 1, heterogeneous objects in IoV are connected by various short and long-range wireless communication technologies to meet the demands of different communication components such as vehicle-to-vehicle (V2V), vehicle-to-road (V2R), vehicle-to-human (V2H), and vehicle-to-infrastructure (V2I).

Vehicle-to-everything (V2X) systems require sensitive instantaneous information such as vehicle speed and coordinates to be exchanged frequently with other vehicles and necessary infrastructures through the wireless medium. Therefore, securing such information exchange is critical to ensure both the normal operation of vehicular systems and the safety of passengers. For example, malicious attackers can passively eavesdrop the communication and hack the passcode to hijack the control unit of vehicles [1], resulting in serious consequence or even death of the passengers. To achieve secure V2X communication, secure key establishment—the procedure wherein two legitimate nodes establish a secret key through a public channel—is highly demanding nowadays. However, traditional key establishment schemes such as public key infrastructure (PKI) and pre-shared key (PSK) are not suitable for IoV due to the following two reasons: 1) it is hard to guarantee the availability of PKI in ubiquitous IoV systems because of the rapid temporal variability and short-time connections of IoV communications; 2) the PSK is not flexible, scalable and can be easily stolen by malicious attackers. Overall, the high mobility and ad-hoc nature of IoV make secure key establishment schemes challenging.

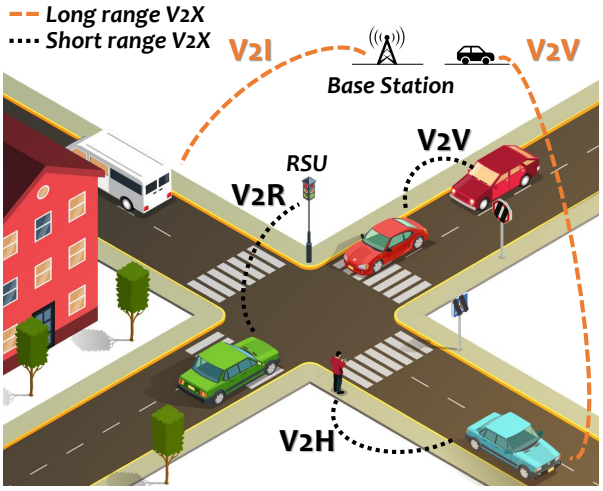


Fig. 1: Long-range and short-range communication scenarios in V2X.

To overcome the aforementioned limitations, physical layer key generation based on channel reciprocity has emerged as a promising technology to establish cryptographic keys for two IoV communication parties. It relies on the principle that the channel characteristics such as Received Signal Strength Indicator (RSSI) observed by the two communication parties will be similar if the communication packets are exchanged during

* Weitao Xu is the corresponding author.

channel coherence time [1] [2]. Existing studies mainly focus on legacy communication technologies such as ZigBee [3], Wi-Fi [2], [4], [5] and 5G [6], whose communication distance is usually in the order of hundreds of meters or even shorter. The requirement of long distance communication prevents many short range wireless communication technologies (e.g., WiFi, Zigbee) from being adopted for secret key generation for IoV systems. LoRa, as one of the prevalent low-power wide-area network (LPWAN) communication technologies, provides a promising solution to long-range IoV secret key generation.

B. Challenges and Contributions

One may ask *"since physical layer key generation has been well studied, why cannot we directly apply existing solutions to LoRa-enabled IoV systems?"* Through revisiting previous secret key generation approaches, we found that existing technologies are not suitable because LoRa-enabled IoV system poses the following two novel challenges for wireless key generation.

1) **Long packet airtime of LoRa.** The key difference between LoRa and legacy communication technologies is that LoRa's long communication distance is achieved by sacrificing its data rate. The data rate of short-range communication technologies such as ZigBee and Wi-Fi is in the order of **kbps** or **Mbps** while the data rate of LoRa can be low as tens of **bps**. The low data rate in turn increases the packet airtime and thus decreases the channel reciprocity because packets cannot be exchanged during the channel coherence time. Hence, the channel characteristics measured by two sides are not similar, which rarely happens for short-range communication technologies. Therefore, traditional approaches developed for Wi-Fi and ZigBee cannot be applied directly in LoRa physical layer key generation.
2) **High mobility of the vehicles.** Unlike the traditional wireless network, the vehicles in IoV are highly mobile, and the communication environment changes rapidly. Therefore, in V2X communication scenarios, the fast fading effect will further exacerbate the above low channel reciprocity problem. However, most existing studies on LoRa physical layer key generation assume the devices are either in fixed location [7] or in low mobility [8], which are not suitable for dynamic IoV scenarios in reality.

A set of pioneering efforts have been made for LoRa-based network [7]–[10]. For example, Han et al. [9] proposed to use multi-bit quantization algorithm and a cascade reconciliation method to generate keys for LoRa-enabled IoV. However, the cascade reconciliation method requires two legitimate nodes to conduct multiple rounds of secret key-related information exchange, which increases the communication overhead and the risk of privacy leakage as well. Gao et al. [10] proposed a novel model-based key generation scheme for LoRa network which has limited key generation rate and is only suitable for static nodes. Therefore, how to establish cryptographic keys effectively and efficiently for long range IoV communications remains an open problem.

To address these challenges, we present a secret key establishment scheme for secure V2X communication called Vehicle-Key. In Vehicle-Key, we propose a novel model that can achieve channel prediction and quantization simultaneously. Moreover, we design an autoencoder-based reconciliation method to correct the mismatches between the keys of legitimate devices. Evaluation in real-world environments shows that Vehicle-Key can achieve a high agreement rate and outperform state-of-the-arts significantly. The main contributions of this paper are as follows:

- We conduct a detailed study to explore the feasibility and challenges of physical layer key generation for LoRa-enabled V2X communication. Based on the findings, we present a secret key establishment scheme Vehicle-Key.
- We propose a novel Bi-directional LSTM (BiLSTM)-based model that can achieve channel prediction and quantization simultaneously. The proposed model addresses the low correlation problem of LoRa channel measurements and hence improves key agreement rate significantly.
- We design an effective autoencoder-based reconciliation method to correct the mismatches between the keys generated by two parties. The proposed method outperforms state-of-the-art methods in terms of both error correction capability and computational cost.
- Extensive real-world experiments show Vehicle-Key can achieve an average key agreement rate of 98.87% and key generation rate of $15\,\mathrm{bits/s}$. Compared to the state-of-the-art, Vehicle-Key improves key agreement rate and key generation rate by 15.10%–49.81% and 9–14$\times$, respectively. Security analysis shows Vehicle-Key is resistant to some common attacks, such as eavesdropping attack and imitating attack. Moreover, we implement the system on a Raspberry Pi and show that Vehicle-Key can be executed in $3.4\,\mathrm{ms}$ while incurring low energy consumption.

The rest of the paper is organized as follows. We present the preliminary results in Sec. II. Then, we discuss the system model and design details in Sec. III and Sec. IV, respectively. Followed by that, we evaluate the Vehicle-Key in Sec. V and discuss the related work in Sec. VI. Finally, we conclude the paper in Sec. VII.

II. Preliminary Study

In this section, we conduct both theoretical and experimental analysis to identify the challenges of physical layer key generation for LoRa-based IoV communications.

A. Theoretical Model

Physical layer key generation is based on the reciprocity property of radio channel that is multipath properties such as gains, phase shifts and delays are identical on both directions of a radio link [11]. Although the radio channel is reciprocal, the channel measurements are not reciprocal due to the following reasons. First, slight time delay on both directions results in measurement inconsistency. Second, even

[1]Coherence time is the time duration over which the channel can be considered to be stable

the transmitter and receiver use the same LoRa module (e.g., SX1278), the hardware imperfection may result in difference in signal measurement. Third, the received signal is mixed by additive noise. Last, the interference power is asymmetric between devices. As stated by Xu et al. [8], the first condition is the main challenge for LoRa-based key generation because of its long packet airtime.

Impact of packet airtime. First, we analyze the impact of packet airtime in LoRa-based IoV communication. Suppose Alice probes the channel at time T_1 and Bob probes the channel at time T_2, then the time delay $\Delta T = T_2 - T_1$ is mainly caused by three factors: transmitting time T_t, propagation time T_p and operation delay T_d. The propagation time is negligible because the transmission speed of radio is $C = 3{\times}10^8\,\mathrm{m/s}$ (the speed of light). For example, suppose the distance between Alice and Bob is $10\,\mathrm{km}$, then the propagation time $T_p = \frac{10000}{3*10^8} = 33.4\ us$. According to our experiments, the hardware operation delay is in milliseconds. Therefore, ΔT is largely dependent on the transmitting time of LoRa transceiver, namely T_t. The transmitting time T_t is further dependent on the bit rate R_b and packet length L: $T_t = \frac{L}{R_b}$. The bit rate of LoRa is $R_b = SF \times \frac{BW}{2^{SF}} \times CR$, where SF denotes spread factor, BW denotes bandwidth and CR denotes code rate. The LoRa radio packet consists of preamble, header, payload and Cyclic Redundancy Check (CRC). The minimum size of a LoRa packet is eight symbols. To achieve long communication distance, the bit rate R_b has to be reduced. Based on different parameter settings, the bit rate of LoRa can be low as hundreds of bps resulting in up to hundreds of milliseconds delay.

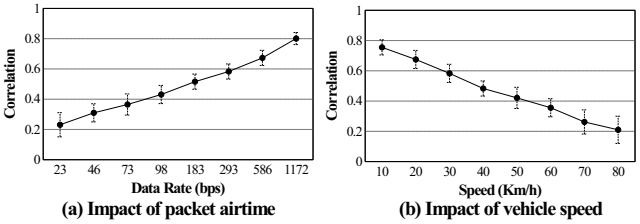
Impact of vehicle's speed. Then, we analyze the impact of vehicle's speed on channel coherence time. Theoretically, the rate at which the radio channel remains stable can be represented by Doppler frequency (f_d) in frequency domain and channel coherence time (T_c) in time domain. Depending on the communication environment, the vehicular wireless communication is modeled differently. Based on the speed of the objects, the channel in IoV communication can be divided into *fast fading channel* and *slow fading channel*.

The fast fading in IoV can be well modeled by Rayleigh distribution [12]. The fast fading channel is suitable for the scenario that there is a large velocity difference between Alice and Bob. In this scenario, the channel gain H should abide by the following probability distribution function (PDF):

$$PDF_H(H,\sigma) = \frac{H}{\sigma^2} e^{-H^2/\left(2\sigma^2\right)}, \tag{1}$$

where δ is an environment-related parameter. In this model, the channel coherence time is calculated by $T_c \approx \frac{0.423}{f_d}$, where the Doppler frequency shift f_d is dependent on the speed between Alice and Bob: $f_d = \frac{|V_A - V_B|}{C} f_0$, where C is the speed of light, f_0 is the carrier frequency, V_A and V_B is the speed of Alice and Bob, respectively.

The slow fading in IoV can modeled by log-normal shadow fading channel [13]. The slow fading model is useful when the relative speed between Alice and Bob is low. In this scenario,



(a) Impact of packet airtime (b) Impact of vehicle speed

Fig. 2: Experimental verification.

the channel gain H is modeled by a log-normal distribution:

$$PDF_H(H,\sigma) = \frac{1}{H\sigma\sqrt{2\pi}} e^{-\frac{\ln(H)}{2\sigma^2}}. \tag{2}$$

If the channel in a slow fading channel does not move over a particular distance (often termed as coherence length L_c), it remains correlated. If we assume the speed of the vehicle is V, the channel coherence time is calculated by $T_c \approx \frac{L_c}{V}$.

From the analysis above, we can see that the speed of the vehicle plays a crucial role in the calculation of coherence time T_c. The higher the speed is, the more frequent the channel changes and the smaller the coherence time is. However, as we mentioned at the beginning of this section, we need to ensure $\Delta T \leq T_c$ so that Alice and Bob can have similar channel measurements. Unfortunately, this condition is not always true due to the low bit rate of LoRa and the high speed of vehicles. For example, if we transmit a 16 bytes packet at $183\,\mathrm{bps}$ ($BW = 125\,\mathrm{kHz}$, $SF = 12$, $CR = 4/8$, $f_0 = 434\,\mathrm{MHz}$), the time delay ΔT is about $700\,\mathrm{ms}$. However, if we assume the speed difference $|V_A - V_B|$ is $40\,\mathrm{km/h}$, the channel coherence time is $27\,\mathrm{ms}$ only, which is significantly lower than ΔT. To sum up, the low data rate of LoRa and high speed of vehicles make physical layer key generation challenging in IoV communication systems.

B. Experimental Verification

To validate the above theoretical analysis, we conducted a comprehensive study in real-world environments using real cars and LoRa modules. We categorize the experiments into two scenarios: rural scenario and urban scenario. In rural scenario, Alice and Bob are placed with a straight long path with line-of-sight (LOS). In urban scenario, two nodes are placed with obstacles (e.g., concrete buildings, etc.). Thus, there is no line-of-sight (NLOS) between them. In each scenario, we explore two V2X applications: V2I (Vehicle to Infrastructure) and V2V (Vehicle to Vehicle). For brevity, we summarize the environment settings as follows:

- Experiment 1: Vehicle to Vehicle in rural.
- Experiment 2: Vehicle to Infrastructure in rural.
- Experiment 3: Vehicle to Vehicle in urban.
- Experiment 4: Vehicle to Infrastructure in urban.

In V2V scenarios (Experiment 1 and 3), two devices are placed on the top of two cars. In V2I scenarios (Experiment 2 and 4), one device is installed on the roof of a building and the other one is placed on the top of a car. The details of the environment and hardware are explained in Section V. The channel physical characteristics used in the experiments are

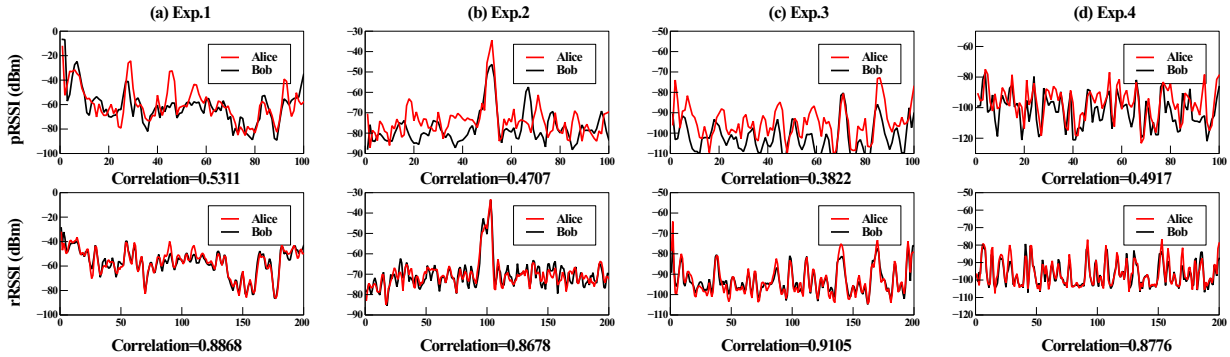


Fig. 3: Preliminary study (Exp.1,2,3,4 represents the experiment under V2V rural, V2I rural, V2V urban, and V2I urban respectively).

RSSI which is the most commonly used metric for LoRa [8], [13]. To quantify the correlation between Alice's channel measurement and Bob's channel measurement, we use Pearson correlation coefficient.

Experiment on packet airtime. To verify the impact of packet airtime, we fix the vehicle speed to be $50\,\mathrm{km/h}$ and calculate the correlation between Alice's RSSIs and Bob's RSSIs by changing the data rates from $23\,\mathrm{bps}$ to $1172\,\mathrm{bps}$. As shown in Fig. 2 (a), the correlation decreases when a lower data rate is used, verifying the previous theoretical analysis that the long packet airtime caused by the low data rate will lead to lower correlation. The correlation drops below 0.6 when the data rate is lower than $293\,\mathrm{bps}$, posing a significant challenge for LoRa-based key generation in IoV scenarios.

Experiment on vehicle's speed. To verify the impact of vehicle speed, we fix the data rate to be $183\,\mathrm{bps}$ and calculate the correlation between Alice's RSSIs and BoB'S RSSIs by changing the speed from $10\,\mathrm{km/h}$ to $80\,\mathrm{km/h}$. As shown in Fig. 2 (b), the correlation decreases when vehicle's speed increases, verifying the previous theoretical analysis that the higher the vehicle's speed is, the lower the correlation is. The correlation drops below 0.6 when the vehicle's speed exceeds $30\,\mathrm{km/h}$, posing another challenge for LoRa-based key generation in IoV scenarios.

C. Our findings

During the experiments, we noticed that the commonly used RSSI is the averaged packet RSSI (pRSSI), which is the average of RSSIs measured during packet reception. The use of pRSSI will result in asymmetry between the two communication parties because the packet airtime in LoRa could be in hundreds of milliseconds or even a few seconds, and the channel conditions may change completely during this relatively long period. We found that the SX127X LoRa transceiver also provides ***register RSSI (rRSSI)***, which is the instantaneous RSSI during packet reception and thus can provide finer granularity.

To verify the feasibility of using rRSSI to improve correlation, we compare RSSI and rRSSI in Fig. 3. The first row of Fig. 3 shows the RSSIs of each experiment and their corresponding correlation coefficients. We can see that the correlation is lower than 0.5 in all the experiments except experiment 1. All of them exist a LOS between two devices.

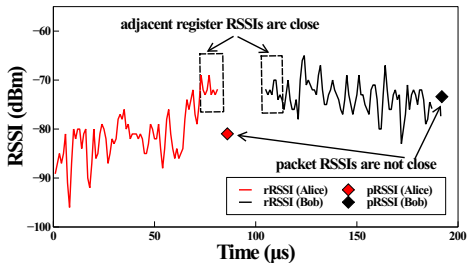


Fig. 4: Packet RSSI vs Register RSSI.

Therefore, the RSSIs measured by Alice and Bob exhibit low correlation because of the low data rate of LoRa and the high mobility of IoV. This finding corresponds to our theoretical analysis above. Then, we plot the rRSSIs of each experiment and their correlation coefficients in the second row of Fig. 3. We can see that the correlation is significantly higher than that of using pRSSI in all experiments.

The advantage of rRSSI over pRSSI can be explained by Fig. 4. First, due to the low data rate, the RSSIs vary greatly during the packet transmission and reception. For example, we can see that the first half part of the samples are different from the second half samples. This phenomenon holds for both Alice's rRSSIs (red color) and Bob's rRSSIs (black color). Therefore, using their average RSSI, namely pRSSI, cannot represent the channel changes during packet reception. Instead, we noticed that the ending part of Alice's rRSSIs is close to the beginning part of Bob's rRSSIs. This is because the timestamps of sampling are close to each other (possibly within channel coherence time). The findings here motivate us to use rRSSI as channel characteristics to generate keys to improve the correlation between Alice and Bob. Second, the number of rRSSI samples that can be generated in one LoRa packet is significantly larger than pRSSI, which indicates that rRSSI can be utilized to improve the key generation rate. However, the instantaneously rRSSI cannot be directly used to generate keys because there still exist some variations as shown in Fig. 4. We propose to employ the mean value of adjacent rRSSI as a new feature for key generation, namely ***adjacent register RSSI (arRSSI)***. To sum up, the preliminary studies not only identifies the challenge but also provide intuitions that can be used to address the challenge. In the followings, we will present system model and design details.

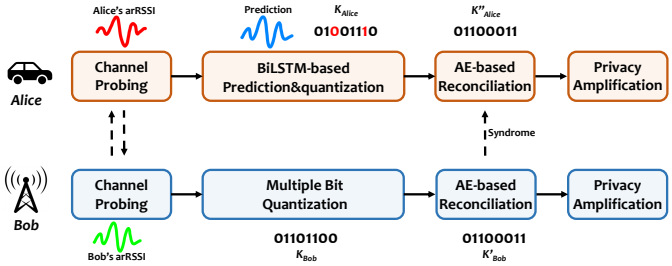


Fig. 5: Vehicle-Key workflow.

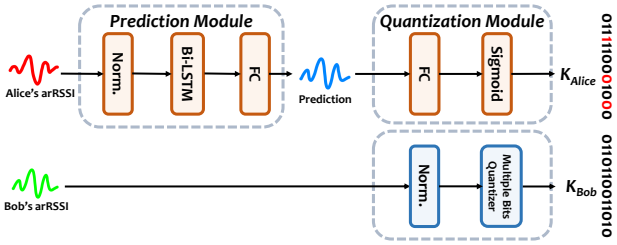


Fig. 6: BiLSTM-based prediction and quantization model.

III. System Model

In this section, we present the system model of Vehicle-Key. We assume that two IoV objects, namely Alice and Bob, intend to generate the same key to secure their communication. One of the objects is a moving vehicle and the other can be moving vehicle, road side unit (RSU) or infrastructure. They are equipped with LoRa communication modules but have no prior shared secrets between each other. We also assume the existence of an attacker Eve who tries to eavesdrop the communication and guess the same key. The LoRa channel reciprocity between Alice and Bob is the basics of Vehicle-Key, i.e., the channel has the same random state if measured in either direction at the same time. As the channel decorrelates in space, the Alice-Eve and Bob-Eve channels are statistically uncorrelated with the Alice-Bob channel. In theory, Eve will measure totally different channel if she is more than $\lambda/2$ away from either Alice or Bob, where λ is the wavelength of the radio waves ($\lambda = 69.12\,\mathrm{cm}$ for $434\,\mathrm{MHz}$ LoRa) [4].

Like many prior works [5], [14], [15], we assume that Eve has the full knowledge of the key agreement protocol and she has the ability to eavesdrop, inject, and replay messages in public channel. Meanwhile, we assume the goal of Eve is to intercept the secret key rather than jamming their communications (i.e., DOS attack). In this paper, we consider two common attacks that are widely used in previous physical layer key generation system [14], [16].

- Eavesdropping attack: Eve tries to eavesdrop on the transmission information of all nodes in the public channel with the aim of generating the same keys.
- Imitating attack: Eve can observe the driving route of Alice or Bob and try to imitate its driving process to obtain similar channel measurements to generate keys.

Eve can also perform Man-in-the-Middle (MITM) attack [17] and reply attack [18]. For these well-studied attacks, we use existing methods which will be discussed in Sec. IV-C. We do not consider other types of attacks such as predictable channel attack [2], [16], where the attacker deliberately move between two devices to cause predictable channel variations. This is because these attacks are usually performed for two static devices but are not suitable for highly mobile vehicles.

IV. System Design

A. Overview

Fig. 5 shows the work-flow of Vehicle-Key. Suppose Alice and Bob are two objects in IoV communication system that need to generate or update a pair of cryptographic key to secure their communication. In the first phase, they measure the channel characteristics by exchanging a number of probe and response packets. Then both Alice and Bob start key generation phase, which includes quantization, reconciliation, and privacy amplification. Finally, the key is used to encrypt/decrypt data to ensure secure communication.

B. Prediction and Quantization

As discussed in Section II, the main challenge is the channel characteristics observed by Alice and Bob are not close to each other because they cannot be measured within channel coherence time. Although the use of arRSSI can increase the correlation between the channel measurement values of Alice and Bob, it is desired that the correlation can be further improved to achieve better system performance. To this end, we propose a BiLSTM-based model to achieve simultaneous prediction and quantization. As shown in Fig. 6, the proposed model consists of prediction module and quantization module. The prediction module is used to predict the channel measurements within channel coherence time while the quantization module is used to convert the predicted arRSSI values into binary bit strings. The input of the model is the arRSSIs observed by one communication party, say Alice. The output is the generated bit sequence. Below we discuss the details.

Prediction module. Since Alice and Bob cannot measure the channel at the same time, our idea is to predict the measurement of one side (say Bob) based on the measurement from another side (say Alice). The prediction module consists of a BiLSTM layer and a fully connected layer. The BiLSTM layer is designed as follows. Each BiLSTM layer contains 32 cells, which is composed of 128 hidden units. The number of hidden units 128 is chosen empirically to prevent overfitting while keeping good performance. We choose the BiLSTM neural networks [19] for prediction due to its superior performance on learning features from sequences with high temporal correlation, which matches the time-varying nature of wireless channel [20]. After the BiLSTM layer, we add one fully connected layer to convert the features extracted by BiLSTM into predicted arRSSI sequence, which is close to Bob's real measurements. Here we only set up one fully connected layer to complete the conversion because BiLSTM learns the overall features rather than the local features of the sequence, and does not need to integrate local features through multiple fully connected layers.

Quantization module. After prediction, both Alice and Bob need to convert the arRSSI values into binary bits. Conventionally, this step is done by adding an individual module called quantization [2]. In Vehicle-Key, we mitigate this extra cost by adding a quantization module after the prediction module in the same network. This module uses sigmoid activation function after a fully connected layer to convert the output matrix of the prediction layer into a binary vector. For Alice, we use sigmoid function because it matches the function of quantizer, i.e., it can map real numbers to the interval (0,1) in a smooth and easy way. The combination of fully connected layer and activation layer can fit a nonlinear transformation, and map the predicted sequence to a 64-bit binary bit space. Using a fully connected layer for mapping can increase the stability of the quantization layer because a small amount of errors in predicted arRSSI sequences have less impact on predicted bit sequence. For Bob, we use the multiple bit quantizer proposed in [2] because it can effectively generate more bits compared to single threshold based methods.

Joint loss function. Combining prediction module and quantization module into a neural network can be more convenient for training, and the joint loss function can be used to optimize the two modules together to reduce extra cost. For the entire network, the joint loss function of the whole network is defined as follows:

$$loss(y, \hat{y}, z, \hat{z}) = \theta \times MSE(y, \hat{y}) + (1-\theta) \times BCE(z, \hat{z}), \quad (3)$$

where y and $\hat{y}$ denote the measured arRSSI values and the predicted arRSSI values, z and $\hat{z}$ denote the binary bit sequences and the predicted sequences of Bob. For our joint task training, we add a hyperparameter θ to balance the weights between BCE and MSE loss to achieve the overall optimal performance. Here, MSE represents Mean Squared Error while BCE denotes Binary Cross Entropy, and they are defined as follows:

$$MSE(y, \hat{y}) = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2, \quad (4)$$

$$BCE(z, \hat{z}) = -\sum_{i=1}^{n} z_i \log \hat{z}_i + (1 - z_i) \log (1 - \hat{z}_i). \quad (5)$$

The principle of the designed loss function is as follows. The proposed model is inherently a multi-task network which serves prediction and quantization purpose. On the one hand, since the aim of Alice is to train a channel prediction model that can predict Bob's arRSSI sequence, we can treat the learning process as a regression problem. In this case, MSE is a commonly used loss function in regression tasks and it can better learn the difference between the measured arRSSI sequence and the predicted arRSSI sequence. On the other hand, the quantization task can be regarded as a classification task with a prediction result of 0 or 1. From this perspective, BCE is a commonly used loss function for binary classification tasks. Therefore, we integrate MSE and BCE into a single loss function by assigning a weight for each function.

The proposed network presents several advantages over previous approaches. Firstly, the prediction module solves the aforementioned low channel reciprocity problem by predicting sequence from the other side. Secondly, the quantization is done seamlessly in the whole network without adding extra modules, which improves the efficiency of training. More importantly, it only needs to be executed on one device such as power-rich devices (e.g., RSU and server). As demonstrated in Sec. V-J, this module only requires 3.38 ms to execute.

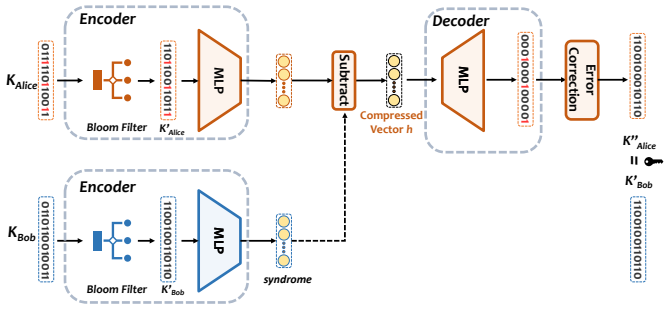


Fig. 7: Autoencoder-based reconciliation method. (Bob only executes the blue part, and then sends the generated syndrome to Alice.)

C. Autoencoder-Based Reconciliation

In practice, we often get $K_{Alice} \approx K_{Bob}$ due to noise. The reconciliation module aims at correcting the bit mismatches between them. While a variety of reconciliation methods have been proposed such as Cascade [21], Error-correction code [22], a recently proposed compressed sensing (CS)-based algorithm [23] has been demonstrated to show better performance. However, it requires multiple iterations in the decoding process which is time-consuming. Recently, deep learning methods such as autoencoder has been employed as new framework of CS [24] to recover images with promising speed and accuracy. Motivated by this, we propose a novel autoencoder-based reconciliation framework to correct the mismatches between Alice and Bob.

As shown in Fig. 7, we design a two-input structure autoencoder for reconciliation. The keys of Alice and Bob are first passed through an adapted Bloom filter [14]. After that, the output of Bloom filter are compressed into a low-dimensional space by a pre-trained encoder. Then the subtraction between the low-dimensional vectors of Alice and Bob will be used as the input of the decoder to calculate the key mismatches. Since Alice and Bob are two independent IoV objects, the process of reconciliation will be executed by both parties separately. Specifically, on Bob's side, he only needs to execute the encoder and send the syndrome code to Alice (the blue part in Fig. 7). On Alice's side, she first executes the encoder, and then calculates the subtraction of the vectors of Alice and Bob which is sent to the decoder to obtain the mismatches between Alice and Bob. The details of the proposed reconciliation method is described below.

The whole model is based on a typical encoder-decoder structure. However, to perform the task of reconciliation, traditional autoencoders [25] cannot be used directly. This is because the autoencoder is designed to compress the input vector X to a shorter code h and reconstruct the input vector X by uncompressing h using decoder, and if one party's key

information is simply compressed and sent using traditional autoencoder, it is easy to be intercepted and cracked by an attacker with information of decoder. To handle this problem, the keys of Alice and Bob are first passed through an adapted Bloom filter [14] to protect the keys against reverse engineering attack. This specially designed Bloom filter can retain position information, which means that its output can retain the same number of mismatched bits as the input key. Although Eve may try to deduce the final key by eavesdropping on probing packets and syndrome, the initial key obtained by eavesdropping is different from the final key generated by the Bloom filter disclosed by syndrome.

Suppose Alice and Bob have generated their initial keys $K_{Alice} \in R^N$ and $K_{Bob} \in R^N$ independently. K_{Bob} and K_{Alice} are used as the input of two different encoders containing Bloom filter and Multilayer Perceptron (MLP). Bloom filters firstly convert the K_{Bob} and K_{Alice} to K'_{Bob} and K'_{Alice}. Then K'_{Bob} and K'_{Alice} pass through MLP f_1 and MLP f_2 to get the code vectors $y_{Bob} = f_1(K'_{Bob})$ and $y_{Alice} = f_2(K'_{Alice})$, where $y_{Alice} \in R^M$ and $y_{Bob} \in R^M$. Here, y_{Alice} and y_{Bob} are high-dimensional condensed expressions of Alice's and Bob's bit sequences, respectively. Then Bob transmits the code vector y_{Bob} to Alice via a public channel.

Suppose y'_{Bob} is the received vector with some noise: $y'_{Bob} = f_1(K'_{Bob}) + e$. Upon receiving y'_{Bob}, Alice calculates the following equation: $h = y'_{Bob} - y_{Alice}$. Note that h can be approximated as the concentrated expression of the mismatches between K'_{Alice} and K'_{Bob}. Transmitting h rather than K'_{Bob} has the following benefits. First, the concentrated expression cannot be used to extract the final keys without Alice's or Bob's original keys (e.g., K_{Alice} and K_{Bob} in Fig. 7). Therefore, it is safe to be transmitted in unauthenticated channel. Second, it is more energy-efficient because the size of vector h is much shorter than K'_{Bob}.

Then, Alice feeds h into a decoder g to obtain the mismatches between K'_{Alice} and K'_{Bob}: $\Delta x = K'_{Alice} \oplus K'_{Bob}$. Afterwards, Alice can correct the mismatches by simply calculating $K''_{Alice} = K'_{\text{Alice}} \oplus \Delta x = K'_{Bob}$, as shown in the Error Correction block in Fig. 7, and finally they can agree on the same key. In the proposed network, we use the following loss function:

$$\arg\min_{f_1, f_2, g} \|\Delta x - (K'_{Bob} \oplus K'_{Alice})\|^2, \tag{6}$$

where Δx represents the decoded mismatches between K'_{Alice} and K'_{Bob}. The loss function is designed in this way so that the distance between the learned mismatches (Δx) and the real mismatches ($K'_{Bob} \oplus K'_{Alice}$) can be minimized.

As mentioned in Sec III, Eve has the ability to modify, insert and replay messages. So two common attacks can be performed by Eve during reconciliation process: MITM and reply attack. Eve can perform MITM attack by impersonating as Alice or Bob during key generation process to modify or insert her messages. To solve this problem, the message authentication code (MAC) method is applied to maintain the integrity of the message during the reconciliation process. Specifically, Bob appends an additional

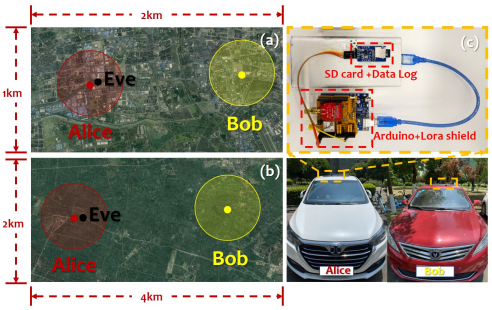


Fig. 8: Experimental setup.

MAC message with y_{Bob}, so the encoded message sent to Alice is $L_{Bob} = \{y_{Bob}, MAC(K'_{Bob}, y_{Bob})\}$. After receiving L_{Bob}, Alice computes $\tilde{K}'_{\text{Alice}}$ and checks its authenticity. If $MAC\left(\tilde{K}'_{\text{Alice}}, y_{Bob}\right) \neq MAC(K'_{Bob}, y_{Bob})$, Alice finds that the message has been modified, indicating there is an attacker Eve. If $MAC\left(\tilde{K}'_{\text{Alice}}, y_{Bob}\right) = MAC(K'_{Bob}, y_{Bob})$, Alice can confirm that this message is from Bob. For replay attacks, we can adopt some commonly used methods such as nounces, timestamps or tagging each message with a session ID [18].

Information reconciliation achieves a higher reliability as evaluate in Sec. V-D, but also reveals part of the information to attackers because Alice and Bob need to exchange some information via public channel. This problem can be solved by using privacy amplification methods such as hash functions [8], [14]. In Vehicle-Key, we apply the commonly used hash function SHA-128 to improve the randomness of the final keys. Then the final keys can be used by symmetric key encryption algorithms such as AES-128 for communications.

V. Evaluation

A. Experimental Setup

1) Data Collection: Fig. 8 shows the experimental setup and data collection process. To evaluate the impact of different devices, we use three different transceivers in the evaluation: Arduino Uno with Dragino LoRa Shield[2] (AVR ATmega328P, SX1278), MultiTech xDot[3] (ARM Cortex-M3, SX1272), MultiTech xDot[4] (ARM Cortex-M3, SX1272). As mentioned in Sec. II-B, we conduct experiments in four different IoV scenarios: V2I-Urban, V2I-Rural, V2V-Urban, V2V-Rural. For each scenario, three identical devices are set up as Alice, Bob and Eve, respectively. In V2V scenarios, all devices are placed on the roof of cars. Alice and Bob travel randomly and the distance between them varies from hundreds of meters to several kilometers. In V2I scenarios, Bob is placed on the roof of a building to emulate an infrastructure device, and Alice is traveling randomly. In both scenarios, we assume the presence of an attacker Eve, who is several meters away from Alice. During data collection, she follows Alice's driving route and tries to generate the same key. A bird view of urban and rural environment is shown in (a) and (b) of Fig. 8, respectively. In total, we collected over 20 hours data and more than

[2]https://www.dragino.com/products/lora/item/102-lora-shield.html
[3]https://www.multitech.com/brands/multiconnect-xdot
[4]https://www.multitech.com/brands/multiconnect-mdot

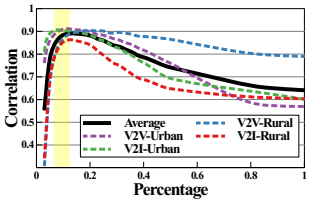


Fig. 9: Impact of arRSSI.

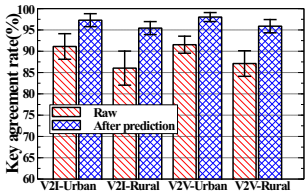


Fig. 10: Impact of prediction.

2,000,000 arRSSI values on different time of different days. The communication parameters used in the experiments are $BW = 125\,\text{kHz}$, $SF = 12$, $CR = 4/8$ and $f_0 = 434\,\text{MHz}$.

2) System Implementation: The implementation details of each network is described as follows. We implement the system on Raspberry Pi 4 which is equipped with 1.5 GHz Quad core Cortex-A72 and 4 GB RAM and connected via LoRa transceivers mentioned in Sec. V-A1. The prediction and quantization model has one layer of BiLSTM and two fully connected layers with 32 and 64 units. The hyperparameter θ of joint loss function is set to 0.9, which is selected through experiments. For the encoder of autoencoder model, each input is connected to a fully connected layer with 32 units, followed by a subtraction layer. In the decoder part, 3 fully connected layers are designed as hidden layers. The units number of the fully connected layers in decoder are experimented in V-D. Unless otherwise stated, we randomly split the whole dataset into three parts: training set (70%), validation set (15%), and test set (15%). The results after 200 epochs are reported.

3) Metrics: As a shared key generation system, we use two commonly used metrics: key agreement rate and key generation rate. For each metric, we report the average result and stand deviation.

B. Evaluation of arRSSI

First, we evaluate the relationship between the arRSSIs with different percentages and their linear correlation. As shown in Fig. 9, as the percentage increases, the correlation between the two arRSSIs first increases and then decreases. This is because when the window is small, the rRSSIs measured by Alice and Bob are within channel coherence time. Thus, the more correlated samples are used, the higher correlation will be obtained. However, when the window is further increased, more measurements beyond channel coherence time are included. As shown in the yellow highlight, the correlation reaches its highest point when about 10% of rRSSI value are used, so we use the 10% rRSSI at the end of Alice/Bob and the beginning of Bob/Alice to form arRSSI.

C. Evaluation of Prediction Module

Then, we evaluate the effectiveness of the proposed prediction model. We compare the key agreement rate with and without our prediction module. As shown in Fig. 10, the proposed method consistently achieves higher agreement rate in all scenarios, indicating our method is effective in improving the channel measurements correlation and hence the agreement rate of the generated bits. Specifically, the key agreement rate can be increased by 5.48%, 11.71% 5.42%, and 10.34% in V2I-Urban, V2I-Rural, V2V-Urban, and V2V-Rural scenarios, respectively. Besides, the standard deviation of the results after prediction module is smaller, which suggests that the proposed method can achieve more stable results.

D. Evaluation of Reconciliation Module

To evaluate the performance of the proposed reconciliation module, we compare it with a state-of-the-art CS-based reconciliation method proposed by Xu et al. [8], [14]. Since they demonstrated that their method is superior to other conventional methods, we only show the result of the CS-based method and our method for the benefit of space. We change the number of units in the hidden layers of decoder and plot the results of different methods in Fig. 11.

It can be observed that the agreement rate of our method using different number of units are higher than that of the CS method. Meanwhile, we notice that the average key agreement rate of the proposed method increases as the number of units increases, indicating that the error correction capability is improved with more units in the hidden layer. The stand deviation of the CS method is higher than our methods, indicating it has low stability. More importantly, our method can reduce the computation cost by $10\times$. The above analysis shows that the proposed reconciliation method has the advantages of high stability, strong error correction capability and low computation cost. We choose AE-64 as the reconciliation method in the following experiments because it achieves a good balance between agreement rate and computational cost.

E. Evaluation of System Robustness

In this subsection, we evaluate the robustness of Vehicle-Key. Specifically, we evaluate the key agreement rate of Vehicle-Key under different devices and speeds.

Impact of different devices. As shown in the Tab. I, the key agreement rates using the three devices are close to each other. On average, Vehicle-Key achieves 99.17%, 98.73%, and 98.73% agreement rate for Dragino LoRa shield, MultiTech xDot, and MultiTech mDot, respectively. The result demonstrates that Vehicle-Key can achieve high agreement rate irrespective of the hardware used.

Impact of different speeds. From Tab. I, we can observe that as the speed increases, the key agreement rates of all three devices drop slightly. Specifically, the key agreement rate drops by 0.36% and 0.64% when the speed increases from $30\,\text{km/h}$ to $60\,\text{km/h}$ and from $60\,\text{km/h}$ to $90\,\text{km/h}$, respectively. However, our system still achieves 99.33%. 98.97%, and 98.33% agreement rate when the speed of the vehicle is $30\,\text{km/h}$, $60\,\text{km/h}$, and $90\,\text{km/h}$, respectively, demonstrating the robustness of Vehicle-Key in different moving speeds.

TABLE I: Agreement rate of different devices and speeds.

Device \ Speed (Km/h)	30	60	90	Mean
Dragino LoRa Shield	99.50%	99.10%	98.90%	99.17%
MultiTech xDot	99.20%	98.90%	98.10%	98.73%
MultiTech mDot	99.30%	98.90%	98.00%	98.73%
Mean	99.33%	98.97%	98.33%	98.87%

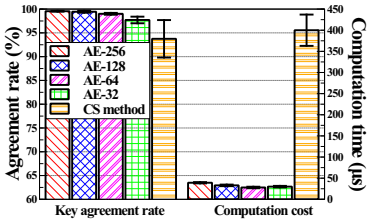


Fig. 11: Impact of reconciliation.

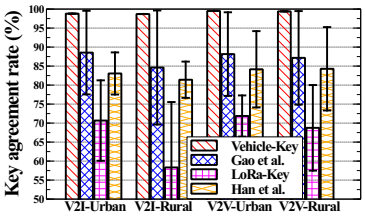


Fig. 12: Comparison of agreement rate.

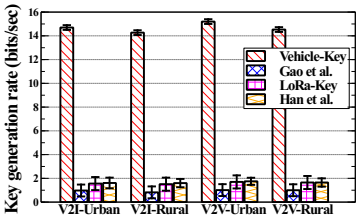


Fig. 13: Comparison of generation rate.

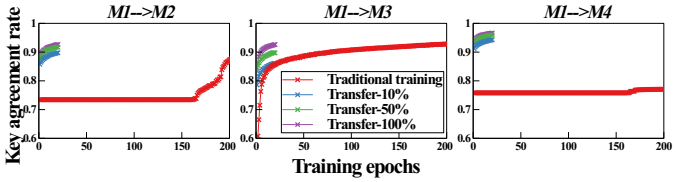


Fig. 14: System generalization analysis.

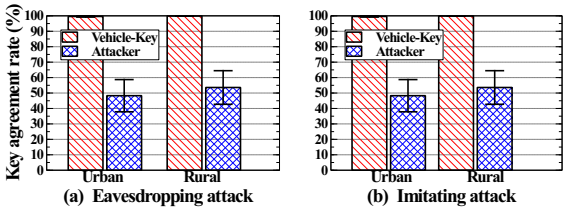


Fig. 15: Security analysis.

F. Comparison with State-of-the-arts

In this section, we compare the proposed method with three representative LoRa-based key agreement approaches, namely, LoRa-Key [8], Han et al. [9] and Gao et al. [10]. We fine-tune their parameters to achieve the best performance for the sake of fairness. Specifically, for Gao et al. [10], the interval and round number are set to 20 and 50 respectively. For LoRa-Key, we set α (the ratio of guard band to data in quantization) to 0.8. The random matrix size of CS-based reconciliation in LoRa-Key and Gao et al. [10] is set to 20×64. For cascade algorithm in Han et al. [9], group length k is set to 3 and the iteration number is set to 4 in our implementation.

The results of different methods in different environments are shown in Fig. 12 and Fig. 13. From Fig. 12, we can see that Vehicle-Key achieves the best key agreement rate in all the scenarios. The average key agreement rate of Vehicle-Key is 49.81% higher than LoRa-Key, 20.48% higher than Han et al. [9], and 15.10% higher than Gao et al. [10]. The stability (standard deviation) of the key agreement rate obtained by Vehicle-Key is also superior to other methods. From Fig. 13, we can observe that the key generation speed of Vehicle-Key is 9× faster than LoRa-Key and Han et al. [9] and 14× faster than Gao et al. [10], which demonstrate the advantage of arRSSI over pRSSI and the superior key agreement ability of Vehicle-Key. To sum up, the results show that Vehicle-Key improves the key generation rate, the key agreement rate significantly compared to the state-of-the-arts.

From the results in Fig. 12 and Fig. 13, we can also see the impact of different environments. The key generation rate in rural areas are lower than that of urban areas because there is less multi-path effect in rural (less buildings and blocks etc.). The key generation rate in V2V scenario is higher than that of V2I scenario because there are more channel variations when both Alice and Bob are moving.

G. Evaluation of System Generalization

In this experiment, we evaluate the performance of Vehicle-Key in new environments. For brevity, we name the models trained in the four scenarios as M1 (V2I-Urban), M2 (V2I-Rural), M3 (V2V-Urban) and M4 (V2V-Rural), respectively. In this experiment, M1 is selected as the base model (the results of other models are similar but are not included due to space limitation). Before applying the model directly to the new environment, we fine-tune the base model with different percentages of training data from the new environment. Transfer-10% means that 10% of the new data is used to train the new model on the base model. The result is obtained by testing the fine-tuned model and traditional trained model on the testing set of the new scenarios.

From the results in Fig. 14, we can see that in the scenarios of $M1 \rightarrow M2$ and $M1 \rightarrow M4$, fine-tuning can make the model converge quickly. Specifically, fine-tuning only needs to use 10% of the data to train 20 epochs to significantly outperform traditional training methods. In the $M1 \rightarrow M3$ scenarios, although the agreement rate of transfer-10% is similar to traditional training after 20 epochs, it saves 90% of the data and 180 training epochs. And the agreement rate of transfer-50% and transfer-100% is still higher than that of the traditional training method after 20 epochs. To sum up, these results demonstrate the proposed model has good generalization ability and can quickly adapt to new scenarios with limited training data.

H. Security Analysis

In this subsection, we analyze the security of Vehicle-Key against the eavesdropping attack and imitating attack mentioned in Sec. III.

1) Against Eavesdropping Attack: In an eavesdropping attack, Eve eavesdrops all the messages between Alice (i.e., vehicle) and Bob (i.e., base station) with the aim of deducing the same key with the eavesdropped information by being

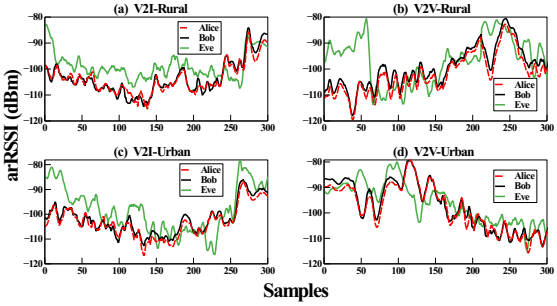


Fig. 16: arRSSI of Alice, Bob and Eve.

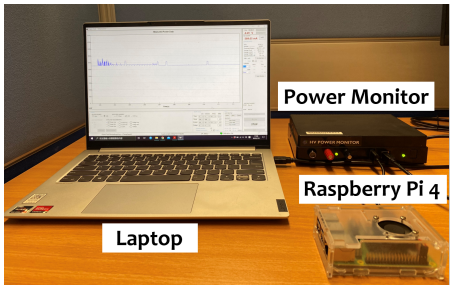


Fig. 17: Power consumption.

statically placed nearby Bob. As mentioned in Sec. IV-C, the only message transmitted between Alice and Bob is y_{Bob}, which is the compressed representation of K'_{Bob}. We now evaluate whether Eve can use y_{Bob} to generate the same key by passing it to the trained decoder. Note that we assume Eve has the full knowledge of the protocol, so she has the same model as Alice and Bob but not their data. As shown in Fig. 15(a), Eve can only reach 50.93% agreement rate in urban environment and 42.18% agreement rate in rural environment. Therefore, even Alice and Bob exchange some information via public channel, the information cannot be used to generate the same key. This is because the exchanged information y_{Bob} is the compressed representation of K'_{Bob}, which infers the difference between Alice's key and Bob's Key but not Eve's key. Feeding y_{Bob} and K_{Eve} into the decoder directly results in wrong corrections and can only achieve about 50% agreement rate as demonstrated in this experiment.

2) Against Imitating Attack: In the imitating attack,Eve can observe the behavior of Alice (i.e., vehicle), and she tries to mimic Alice's driving route with the aim of generating a similar arRSSI sequence. Fig. 16 shows the arRSSI traces of Alice, Bob and Eve in different environments. We can see that the overall pattern of Eve is similar to Alice and Bob, but the small-scale variations are totally different. This is because the channel variations are mainly caused by three factors: path loss, shadow fading, and small-scale fading. If Eve follows Alice's driving route, she can obtain similar path loss and shadow fading, but she cannot observe similar small-scale fading due to multi-path effect, which is the main source of randomness [2], [11]. Fig. 15(b) shows the key agreement rate of Eve in urban and rural environments. We can observe that the key agreement rate of Vehicle-Key is 98.96% and 99.15% in urban and rural environment, while the agreement rate of Eve is only 48.28% and 53.59% in urban and rural environments, respectively.

TABLE II: NIST test.

NIST Test	p-value
Frequency	0.209510
DFT Test	0.708206
Longest Run	0.710984
Linear Complexity	0.398762
Block Frequency	0.925826
Cumulative Sums	0.375082
Approximate Entropy	0.140715
Non Overlapping Template	0.497892

As mentioned above, Eve can at most achieve approximately 50% agreement rate, which means if we use 128-bit key for encryption, the probability of deducing the same key is extremely low, i.e., $0.5^{128} = 2.94e^{-39}$. Therefore, the fading nature of the wireless channel guarantees the security of Vehicle-Key. Eve cannot generate the same key as long as she is half wavelength (i.e., $34.56\,\mathrm{cm}$ for $434\,\mathrm{MHz}$ LoRa) away from legitimate devices, which is a realistic assumption in real-world IoV scenarios.

I. Key Entropy and Randomness

The NIST suite of statistical tests [26] is used to validate the randomness of the generated keys. This suite outputs p-values to indicate the randomness of the key sequence. Conventionally, if p-value is less than 1%, the randomness hypothesis is rejected which implies the secret key is not random. From Tab. II, we can see that all the p-values of different tests are higher than 1%, indicating the generated keys pass the random test and have high randomness.

J. Power Consumption

As shown in Fig. 17, we use power monitor to evaluate the computation time and energy consumption required by Vehicle-Key to generate a 128-bit key. The computation time and energy consumption of different component is shown in Tab. III. Note that Alice and Bob perform different steps and hence have different results. Moreover, the computation time of privacy amplification is in the order of microsecond, and hence is not included in Tab. III. The results show that the key generation can be completed in $3.4\,\mathrm{ms}$ and incur low energy consumption on both Alice and Bob.

TABLE III: Power consumption.

	Computation time (ms)		Energy consumption (mJ)	
	Alice	Bob	Alice	Bob
Prediction and quantization	3.38	0.42	12.8947	1.44
Reconciliation	0.0308	0.0077	0.1113	0.0278
Total	3.4108	0.4277	13.006	1.4678

VI. Related Work

Physical layer security has attracted considerable attention in the past decades. A large majority of prior work focus on legacy wireless communication technologies such as ZigBee and Wi-Fi. For example, Wang et al. [27] proposed a secret key

generation method by employing the consistently distributed phase of channel responses. Xi et al. [5] proposed an authentication and key agreement scheme by using the channel state information (CSI) of mobile devices. Liu et al. [28] employed the channel response information from multiple Orthogonal Frequency-Division Multiplexing (OFDM) subcarriers to provide fine-grained channel characteristics.

With the popularity of LoRa, researchers started to study the key generation problem in LPWAN. Ruotsalainen et al. [7] investigated the impact of different LoRa configurations on the performance of key generation, including different spreading factors, bandwidths, and environments. Xu et al. [8] conducted the first comprehensive feasibility study and designed a complete LoRa key generation protocol which is named LoRa-Key. Gao et al. [10] proposed a model-based key generation system for LoRa networks. However, the LoRa devices used in these studies are assumed to be static and hence the solutions are not suitable for mobile IoV network. The researchers in [9] designed a key generation systems for LoRa-based IoV systems independently. However, they directly apply existing methods and the evaluation is only conducted in limited environments.

VII. Conclusion

This paper studies the wireless key generation problem in LoRa-enabled IoV scenarios. We propose a key generation scheme for LoRa-enabled IoV communications which is named Vehicle-Key. In Vehicle-Key, we propose a BiLSTM-based prediction and quantization model that utilizes arRSSI to solve the low channel reciprocity problem. Additionally, we propose an autoencoder-based reconciliation approach to correct the mismatched keys. Extensive real-world evaluation using three different types of LoRa devices shows that Vehicle-Key can achieve an average key agreement rate of 98.87% for two LoRa-enabled IoV objects and outperforms the state-of-the-arts by 15.10%–49.81%. Meanwhile, security analysis demonstrates Vehicle-Key is secure against several common attacks. We also implement the system on Raspberry Pi and show that Vehicle-Key can generate a 128-bit key in $3.4\,\mathrm{ms}$.

Acknowledgment

The work described in this paper was substantially sponsored by the project 62101471 supported by NSFC and was partially supported by the Shenzhen Research Institute, City University of Hong Kong. The work described in this paper was partially supported by Shenzhen Science and Technology Funding Fundamental Research Program (Project No. 2021Szvup126), NSF of Shandong Province (Project No. ZR2021LZH010), Hong Kong RGC ECS grant (Project No. CityU 21201420), and a grant from Chow Sang Sang Group Research Fund sponsored by Chow Sang Sang Holdings International Limited (Project No. 9229062).

References

[1] H. Hasrouny, A. E. Samhat, C. Bassil, and A. Laouiti, "Vanet security challenges and solutions: A survey," *Vehicular Communications*, 2017.

[2] S. Jana, S. N. Premnath, M. Clark, S. K. Kasera, N. Patwari, and S. V. Krishnamurthy, "On the effectiveness of secret key extraction from wireless signal strength in real environments," in *Mobicom*, 2009.

[3] T. Aono, K. Higuchi, T. Ohira, B. Komiyama, and H. Sasaoka, "Wireless secret key generation exploiting reactance-domain scalar response of multipath fading channels," *IEEE Trans. Antennas Propag.*, 2005.

[4] S. Mathur, W. Trappe, N. Mandayam, C. Ye, and A. Reznik, "Radio-telepathy: extracting a secret key from an unauthenticated wireless channel," in *Mobicom*, 2008.

[5] W. Xi, C. Qian, J. Han, K. Zhao, S. Zhong, X.-Y. Li, and J. Zhao, "Instant and robust authentication and key agreement among mobile devices," in *CCS*, 2016.

[6] L. Jiao, N. Wang, P. Wang, A. Alipour-Fanid, J. Tang, and K. Zeng, "Physical layer key generation in 5g wireless networks," *IEEE Wirel Commun*, 2019.

[7] H. Ruotsalainen, J. Zhang, and S. Grebeniuk, "Experimental investigation on wireless key generation for low-power wide-area networks," *IEEE Internet Things J.*, 2019.

[8] W. Xu, S. Jha, and W. Hu, "Lora-key: Secure key generation system for lora-based network," *IEEE Internet Things J.*, 2018.

[9] H. Biao, P. Sirui, W. Celimuge, W. Xiaoyan, and W. Baosheng, "Lora-based physical layer key generation for secure v2v/v2i communications," *Sensors*, 2020.

[10] J. Gao, W. Xu, S. Kanhere, S. Jha, J. Y. Kim, W. Huang, and W. Hu, "A novel model-based security scheme for lora key generation," in *IPSN*, 2021.

[11] N. Patwari, J. Croft, S. Jana, and S. K. Kasera, "High-rate uncorrelated bit extraction for shared secret key generation from channel measurements," *TMC*, 2010.

[12] M. K. Simon and M.-S. Alouini, *Digital communication over fading channels*, 2005.

[13] J. Wan, A. Lopez, and M. A. A. Faruque, "Physical layer key generation: Securing wireless communication in automotive cyber-physical systems," *TCPS*, 2018.

[14] W. Xu, Z. Li, W. Xue, X. Yu, B. Wei, J. Wang, C. Luo, W. Li, and A. Y. Zomaya, "Inaudiblekey: Generic inaudible acoustic signal based key agreement protocol for mobile devices," in *IPSN*, 2021.

[15] H. Liu, J. Yang, Y. Wang, Y. J. Chen, and C. E. Koksal, "Group secret key generation via received signal strength: Protocols, achievable rates, and implementation," *TMC*, vol. 13, no. 12, pp. 2820–2835, 2014.

[16] H. Liu, Y. Wang, Y. Ren, and Y. Chen, "Bipartite graph matching based secret key generation," in *INFOCOM*, 2021.

[17] S. Eberz, M. Strohmeier, M. Wilhelm, and I. Martinovic, "A practical man-in-the-middle attack on signal-based key generation protocols," in *ESORICS*, 2012.

[18] S. Malladi, J. Alves-Foss, and R. B. Heckendorn, "On preventing replay attacks on security protocols," IDAHO UNIV MOSCOW DEPT OF COMPUTER SCIENCE, Tech. Rep., 2002.

[19] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm and other neural network architectures," *IJCNN*, 2005.

[20] T. Plötz and Y. Guan, "Deep learning for human activity recognition in mobile computing," *Computer*, 2018.

[21] G. Brassard and L. Salvail, "Secret-key reconciliation by public discussion," in *EUROCRYPT*, 1993.

[22] H. Liu, J. Yang, Y. Wang, and Y. Chen, "Collaborative secret key extraction leveraging received signal strength in mobile wireless networks," in *INFOCOM*, 2012.

[23] Q. Lin, W. Xu, J. Liu, A. Khamis, W. Hu, M. Hassan, and A. Seneviratne, "H2b: Heartbeat-based secret key generation using piezo vibration sensors," in *IPSN*, 2019.

[24] H. Wu, Z. Zheng, Y. Li, W. Dai, and H. Xiong, "Compressed sensing via a deep convolutional auto-encoder," in *VCIP*, 2018.

[25] A. P. S. Chandar, S. Lauly, H. Larochelle, M. M. Khapra, B. Ravindran, V. Raykar, and A. Saha, "An autoencoder approach to learning bilingual word representations," in *NIPS*, 2014.

[26] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, and E. Barker, "A statistical test suite for random and pseudorandom number generators for cryptographic applications," DTIC Document, Tech. Rep., 2001.

[27] Q. Wang, H. Su, K. Ren, and K. Kim, "Fast and scalable secret key generation exploiting channel phase randomness in wireless networks," in *INFOCOM*, 2011.

[28] H. Liu, Y. Wang, J. Yang, and Y. Chen, "Fast and practical secret key extraction by exploiting channel response," in *INFOCOM*, 2013.