Using Wireless Link Dynamics to Extract a Secret Key in Vehicular Scenarios

Xiaojun Zhu, *Member, IEEE*, Fengyuan Xu, *Member, IEEE*, Edmund Novak, Chiu C. Tan, *Member, IEEE*, Qun Li, *Senior Member, IEEE*, and Guihai Chen, *Member, IEEE*

Abstract—Securing a wireless channel between any two vehicles is a crucial component of vehicular networks security. This can be done by using a secret key to encrypt the messages. We propose a scheme to allow two cars to extract a shared secret from RSSI (Received Signal Strength Indicator) values in such a way that nearby cars cannot obtain the same key. The key is information-theoretically secure, i.e., it is secure against an adversary with unlimited computing power. Although there are existing solutions of key extraction in the indoor or low-speed environments, the unique channel conditions make them inapplicable to vehicular environments. Our scheme effectively and efficiently handles the high noise and mismatch features of the measured samples so that it can be executed in the noisy vehicular environment. We also propose an online parameter learning mechanism to adapt to different channel conditions. Extensive real-world experiments are conducted to validate our solution.

Index Terms—Vehicular network, security, privacy, secret key extraction, RSSI, measurement

1 Introduction

Vehicular networks have attracted much research effort recently given their potential in improving public safety and traffic management [1], [2]. Through vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) connectivity, important information such as road conditions (e.g., construction) and traffic conditions (e.g., traffic is moving slowly ahead) can be transmitted to allow drivers and their vehicles to make the appropriate decisions. In vehicular networks, security is an important component and one of the most fundamental security requirements is the ability to establish a secure channel between two arbitrary cars. Rather than relying on a public key infrastructure (PKI) based solution [3], in this paper, we propose an alternative approach to allow two arbitrary cars to establish a secret key for secure communications. Our approach can be used in environments where a PKI has not been established.

This is done by having both cars continuously sample the wireless link and then extract a shared secret key based on the signal strength fluctuations. Since the fluctuations are resulted from the unpredictable wireless channel dynamics, they cannot be observed by an adversary at a distance more than half the wavelength of the wireless signal (e.g., that is 6.25 cm for 2.4 GHz wireless channel) so that the key is secure. Similar to existing works [4], [5], [6], [7], the extracted key is information-theoretically secure, in contrast to the classic key establishment method such as Diffie-Hellman that relies on computational hardness assumption. To sample channel dynamics, two parties capture RSSI values by sending packets back and forth to each other. This will require addressing the following challenges that have rendered the previous approaches [4], [5], [6], [7] unusable.

First, vehicular environments have very short *channel coherence time*, the time duration for which the wireless channel remains unchanged, due to rapid environment change. Measurements [8], [9] have shown that channel coherence time in vehicular environments can be as short as a few hundred microseconds. Although short coherence time will give high randomness for key extraction in low speed mobile environments [4], [5], [6], [7], very short coherence time in vehicular environments will pose a big challenge. Because, to sample the same channel dynamics, a pair of sample packets must be sent by two parties (vehicles) respectively within a duration of coherence time. Due to the half-duplex nature of current wireless platforms, however, we find that the round-trip time of a wireless packet may be longer than the coherence time. Applying existing solutions to vehicular environments results in unsatisfactorily slow key generation.

Second, RSSI is widely available among off-the-shelf 802.11 radios so that our solution can be readily implemented on existing wireless platforms without hardware changes. But RSSI has poor accuracy in characterizing the channel condition, compared to the Channel Impulse Response (CIR) measurements [4]. We refer to both the

- *X. Zhu is with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China, and the Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing 210023, China. E-mail: xzhu@nuaa.edu.cn.*
- *F. Xu is with the State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China. E-mail: fengyuan.x@gmail.com.*
- *E. Novak and Q. Li are with the Department of Computer Science, College of William and Mary, Williamsburg, VA 23187. E-mail: ejnovak@email.wm.edu, liqun@cs.wm.edu.*
- *C.C. Tan is with the Department of Computer and Information Sciences, Temple University, Philadelphia, PA 19122. E-mail: cctan@temple.edu.*
- *G. Chen is with the State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China, and the Shanghai Key Laboratory of Scalable Computing and Systems, Shanghai Jiao Tong University, Shanghai 200240, China. E-mail: gchen@nju.edu.cn.*

Manuscript received 3 Feb. 2015; revised 4 Oct. 2015; accepted 2 Apr. 2016. Date of publication 25 Apr. 2016; date of current version 1 June 2017.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TMC.2016.2557784

short-coherence-time effect and the RSSI error as noise due to the inability to distinguish them.

Therefore, one challenging issue is how to reduce the effect of noise in captured RSSI traces without eliminating too much randomness in them. After filtering out the slow variation [4], which is caused by distance changes, we observe that the noise is very strong, comparable to the level of fluctuation due to the channel dynamics. For high quality traces considered by previous research, the noise level is negligible compared to the signal fluctuation, which is not the case in our trace. Thus, we have a dilemma. If we keep a large portion of the fluctuation, i.e., do not filter out noise sufficiently, the mismatches of bits at the two sides will become so numerous that the number of final extracted bits will be zero after subtracting the effort to correct such mismatches. On the other hand, if we remove too much of the fluctuation, even though mismatches are reduced, the final bits will contain little randomness. Therefore, the main obstacle of this paper is to delineate the fine line between noise and fluctuation of the channel dynamics.

To the best of our knowledge, we are the first to consider key extraction in vehicular environments using RSSI values. None of the previous methods [4], [5], [6], [7] can be used directly in extracting secret bits in vehicular environments. By borrowing ideas from [4], [5], [6], [7], we address challenges arising from vehicular networks as follows (A preliminary version of this paper appeared in [10]).

- We propose weighted sliding window smoothing to reduce noise (Section 4). This smoothing method considers both local white noise and mismatched sensing time, and can reduce the strong noise in vehicular environments better than existing smoothing methods used in the key extraction literature, which has been verified in our experiments.
- We propose a systematic way to ensure the randomness of the resulting key (Section 5). We discover that a Markov model is appropriate to capture the dependency among random bits, based on which we select a randomness extractor to produce perfect random bits. Besides ensuring the randomness of the resulting key, we utilize more mutual information among captured samples, compared to existing randomness extraction methods relying on sub-sampling or random hashing. The extracted bits pass the NIST test [11].
- We propose an online parameter learning scheme to adaptively adjust parameters which offers more steady performance in different environments (Section 6). To learn the parameters in real-time, we propose a light-weight objective function for training parameters, which performs comparably to the original objective but incurs much less computation.
- Our experiments are conducted using data collected from real world vehicular environments (Section 7).

2 Related Work

Extracting secret keys from the unpredictable radio channel variations has gained considerable research interests recently. Azimi-Sadjadi et al. [12] propose an extraction scheme based on signal envelopes and evaluate it on a Rayleigh fading channel model. Later both Mathur et al. [4] and ASBG [6] leverage the channel's level crossings to generate secret keys shared between two parties in the indoor and low-speed outdoor environments respectively. In environments where the channel does not have sufficient variations, key extraction can be very slow due to lack of randomness. Different approaches are proposed to address this issue [13], [14], [15]. Gollakota and Katabi [13] propose iJam for OFDM-based systems. In iJam, the receiver randomly jams the signal in such a way that it can still decode the message while the adversary cannot. Also targeting at OFDM systems, Liu et al. [14] propose to use the channel response from multiple subcarriers simultaneously to speed up key extraction. Huang and Wang [15] attach two antennas to the transmitter, and, by controlling amplitude and phase of each symbol on each antenna, they introduce channel fluctuations and increase key extraction rate. Some key extraction approaches [7], [16] are proposed specifically for the mote platform.

Other physical properties can also be used for key extraction (e.g., [17]). Recently, Safaka et al. [18] utilize the phenomenon of random packet loss to establish pairwise secrets for a group of wireless nodes. References [19], [20] target at extracting a group key, instead of pairwise keys, for multiple devices. There has also been much work on protecting WLANs and sensor networks [21], [22], [23].

Along with these approaches are security concerns in the literature [6], [24], [25]. In a static environment, the adversary can create predictable channel variations useful for breaking the key, by using an object to repeatedly block and unblock the line-of-sight path between the sender and receiver [6]. Such attacks are hardly practical in dynamic environments such as vehicular scenarios since the adversary cannot control all channel variations, which is different from the scenario in [6]. Reference [24] conducts experiments using MICAz sensor motes, and finds that the key extracted by level crossing [4] may have up to 81.97 percent in common with that obtained by colluded adversaries at a distance less than 6 cm away from the transmitter. However, when the distance increases to 90 cm, the in-common percentage decreases to less than 57 percent, only slightly better than random guessing. This security concern may not apply to vehicular environments, since the adversaries are at least several meters away from both the transmitter and receiver. Reference [25] suggests that key extraction schemes might be vulnerable to man-in-the-middle-attack. We do not consider this attack and leave it as an important future work.

Neither of the above works considered key extraction in vehicular networks, where the radio channel has unique characteristics. In the technical perspective, we have to address two issues: sensing value discrepancy at the two parties due to noise, and dependency among the quantized bits that weakens the security of the key. Similar issues are also discussed in other testbed-based key extraction works [4], [5], [6], [7], [14], but unfortunately the methods applied in prior work are insufficient or inappropriate in vehicular case due to strong noise. To reduce noise, we find a new technique that outperforms existing smoothing techniques such as sliding window smoothing used in [14] and cubic Farrow filter used in [7], [19] with the comparison result presented in Section 7. For dependency elimination, the sub-sampling method in [4], [5] discards many bits along with useful mutual information. That information is precious to our key extraction scheme due to short window of channel sampling time. HRUBE [7] tries to utilize all samples through decorrelation process. The main drawback is that the samples and the resulting key, though uncorrelated, may still be dependent. In contrast, we analyze the dependency among bits and remove it by splitting the bit sequence accordingly into independent sub-sequences. Therefore we do not waste captured channel information nearly as much.

Additionally, we propose using a deterministic randomness extractor for entropy condensing, rather than the random hashing scheme applied in the previous work [6], where one uniformly and randomly chooses a hash function from a (strongly) two-universal hash family and the hashed value is taken as the extracted random bits [26]. There are two reasons. First, theoretically, such random hashing based approaches extract less bits than our approach, since their efficiency (the number of output bits to the number of input bits) is upper bounded by the *min-entropy* of the input bits [26], which is smaller than or equal to Shannon entropy, the efficiency upper bound of our approach. Second, practically, it is hard to verify whether the randomness of their extracted bits is from wireless channel. Even if the input bits are not random, the hash value is random due to randomly chosen hash functions. Consequently, it is inappropriate to use the NIST test tool for randomness validation. For the same reason, the NIST test in this work is conducted on the extracted bits *before* privacy amplification that also relies on random hashing, in contrast to previous works [14], [19]. Our work borrows results from the literature of randomness extraction [26], [27], quantum key distribution [28], [29] and optimization [30], which will be introduced where they are used.

3 Background

We consider a passive threat model in this paper, where the adversary can eavesdrop transmitted messages, and cannot actively jam normal transmissions. We focus on extracting a shared secret key for two moving cars. We assume that the two cars, Alice and Bob, have a wireless channel to exchange messages when they meet, and they have the ability to record the RSSI reading of the exchanged messages. The adversary, Eve, has complete knowledge about the procedure and the exchanged messages. The goal of the system is to extract a shared secret key for Alice and Bob in such a way that Eve cannot infer any information about the key. Similar to [6], [7], [14], [18], we do not consider the authentication problem, which is also an important research direction (e.g., [31]).

The general key establishment process is as follows. Here, we assume Alice initiates the process. Alice keeps sending indexed probes to Bob, and Bob immediately returns back acknowledgements (ACKs) upon reception. The probes and ACKs are made as short as possible to deal with the short-coherence-time issue. Both sides record the RSSI reading of received packets along with their indices. The RSSI readings observed by Alice and Bob are denoted by $X = (x_1, x_2, \dots, x_n)$ and $Y = (y_1, y_2, \dots, y_n)$ respectively. The random bits are then extracted from these readings. We also refer to X and Y as *raw data* or *raw readings*. A classic method for key extraction is level crossing [4]. Next, we will introduce the principals behind level crossing and its limitations, followed by our techniques to address these limitations.

3.1 Level Crossing: Overview and Limitations

Level crossing consists of two steps. First, Alice and Bob keep probing the channel and collecting RSSI readings. They map each reading to a temporary bit as follows. Consider the case for Alice. Let μ_X be the mean of X, and let σ_X be the standard deviation. Each reading x is mapped to a temporary bit via a quantizer Q such that Q(x)=1 if $x > q_+$; Q(x) =0 if $x < q_-$; Q(x) =e, otherwise. Here e is an undefined state and $q_+ = \mu_X + \alpha\sigma_X$, $q_- = \mu_X - \alpha\sigma_X$ where α is a parameter to be tuned. This can be seen in Fig. 2. Bob's quantizer is similar.

Second, Alice and Bob communicate to get the final bits from the temporary bits. They identify *excursions* in the temporary bits. Excursion is a consecutive string of 1's or 0's with length at least m (where m is the second parameters to be tuned). For example, if the temporary bits are "$e0000111e111e$" and $m = 3$, then there are three excursions: $e\,\overbrace{0000}\,\overbrace{111}\,e\,\overbrace{111}\,e$. Alice finds from her temporary bits all excursions, and sends the indexes of all excursion centers to Bob. On receiving the indexes, Bob checks each index to see whether he also has an excursion around this index, and sends the result back to Alice. Finally, they quantize each common excursion to a bit. For the above example, if Bob has the same temporary bits, then they both get 011. We call the final bits (a bit vector) the *quantized bits*. The bits at Alice and Bob may be different. Each different bit is a *mismatch*. The ratio of the number of mismatches and the number of the quantized bits is defined as *mismatch rate*.

The level crossing algorithm subtracts a windowed moving average from the raw data to reduce the influence of slow variation, where the output of the sliding window is the slow variation. This creates the third parameter: the window size s. The residuals, rather than the raw readings, are fed into the quantizer Q. In summary, we have three parameters: reading threshold α, excursion threshold m, and window size s.

To determine the effectiveness in a vehicular environment, we implemented the two most relevant methods, level crossing [4] and ASBG [6], [32]. Due to the noise in a vehicular environment, the ASBG method applied to our trace leads to a very high mismatch ratio[1], which renders the secret extraction rate not satisfactory. Our experiments find that the generic level crossing method [4] can result in a relatively low mismatch ratio compared to the ASBG method. Therefore, we mainly improve on the generic level crossing method. However, directly applying level crossing is not sufficient to achieve good bit rate. Even though level crossing is a groundbreaking approach to secret extraction, it still has some unsolved problems. Applying it to our vehicular trace shows the following.

First, we observe high noise level in vehicular environments, compared to low noise level in low-speed environments [4], [5], [6], [7]. To illustrate, we plot 500 raw readings from a collected vehicular trace in Fig. 1 (The process of trace collection is elaborated in Section 7). We can see that the noise level is high. Indeed, the residuals (after subtracting slow variations from raw readings) for this trace have a

[1] For single-bit extraction scheme with a block size of 50, the mismatch ratio of ASBG on a typical trace is over 44 percent.

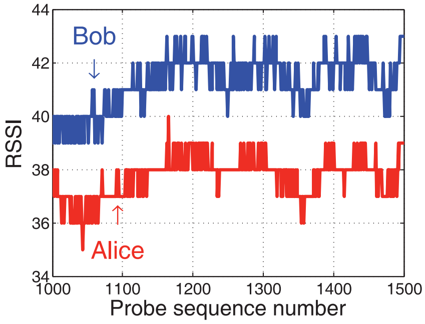


Fig. 1. High noise in vehicular environment (1-second period). RSSI of Alice changes in a different way as Bob. When Alice goes 1db down, Bob usually remains the same, or even goes 1 db up.

low correlation coefficient of 0.52. We have to design schemes to reduce the noise; otherwise it causes mismatches that seriously influence the performance.

Second, there are no guidelines to select suitable parameters for level crossing. In our experiments we notice that the performance is very sensitive to parameter setting. Setting parameters, however, is hard due to the noise level comparable to channel dynamics. A fine line must be found which can separate the noise from the signal, in order to maintain a minimal mismatch rate.

Third, the level crossing scheme does not estimate the randomness of the generated bit string nor does it generate a bit string that is necessarily random. We show one such case in Fig. 2. In the figure, the quantized bits are not random (readings are consecutively below the lower threshold or over the upper threshold). They are dependent. The original paper [4] employs subsampling on quantized bits to counteract dependency. We find, however, random subsampling can discard many important bits containing high randomness, resulting in slow key generation. Instead, we aim to find an optimal and controllable method to remove dependency among the generated bits.

Fourth, the level crossing scheme uses a fixed set of parameters, which does not adapt to the drastic change of channel dynamics in vehicular environments. Fig. 3 shows the correlation coefficient of residual readings at different time periods (2-second units). We observe that the correlation varies at different times, which suggests that a fixed set of parameters may not work well.

3.2 Overview of Our Approach

We organize our solution into the framework in Fig. 4. In the framework, information reconciliation and privacy amplification are borrowed from [5], [6], [32]. Our contribution consists of the steps shown in the solid line boxes. We overview the whole process in this section.

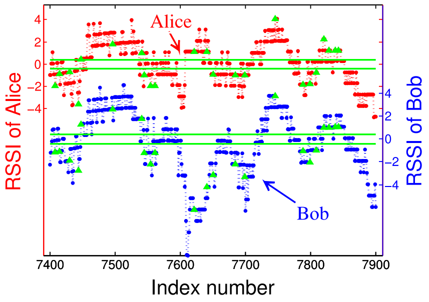


Fig. 2. Level crossing on a portion of residuals. The two lines for alice (top) or bob (bottom) are the q_+, q_- in quantizer Q, and each triangle corresponds to a quantized bit.

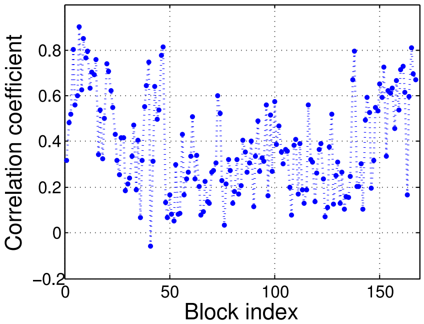


Fig. 3. Non-stable channel condition. Each block corresponds to 2-second residuals. Correlation coefficient reflects the noise level and it changes over time.

Depending on how level-crossing's parameters are selected, we propose two schemes: a fixed parameter scheme (or fixed scheme), and an online parameter learning scheme (or online scheme). The fixed scheme is the basic scheme, where all parameters are determined beforehand. Derived from the fixed scheme, the online scheme adjusts some parameters in an online fashion in order to adapt to different environments. We introduce the fixed scheme and briefly mention the online scheme.

Alice and Bob continuously get RSSI readings by periodically sending probing messages. These readings are smoothed in real-time. Our smoothing method, different from existing ones, tries to maximize the correlation coefficient of the smoothed readings. Section 4 details the design and exact procedure. The smoothed readings are fed into the level crossing algorithm, which produces quantized bits.

The next step is information reconciliation, which addresses the issue that the quantized bits at Alice and Bob may be different. The task of correcting these different bits, i.e., mismatches, is referred to as information reconciliation in the quantum key distribution literature. Following [6], we choose to implement Cascade [29], the *de facto* information reconciliation method, due to simplicity and efficiency. During the execution of Cascade, parity bits exchanged are exposed to the adversary. This exposed information will be reduced by privacy amplification so that Eve has less than 1 bit information about the key on average [28].

Then, Alice and Bob have the same quantized bits (otherwise, they will restart the process). But the bits may not be secure due to dependency. To address this issue, Alice (Bob) extracts random bits from her (his) own quantized bits. This is done by using a deterministic randomness extractor based on a Markov model of the quantized bits. We present the model as well as the extractor in Section 5. In our experiments, the randomness of the extracted bits is evaluated by the NIST test [11], the state-of-the-art statistical test tool for randomness.

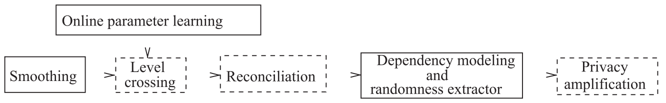

Online parameter learning
Smoothing
Level crossing
Reconciliation
Dependency modeling and randomness extractor
Privacy amplification

Fig. 4. Flowchart of the system. We focus on the three steps within solid boxes, each of which is a section in this paper.

Now the two sides have perfect random bits, but the bits are still not secure because the information reconciliation step leaks some information. Similar to [6], we apply privacy amplification [28] to distill such information. Specifically, Alice randomly chooses a strong, 2-universal hash family, according to the amount of leaked information. From this family, Alice randomly chooses a hash function, which she sends to Bob. Then they hash the random bits with this hash function and use the hash value as the key, about which Eve has less than one bit information on average. According to [28], wiping out Eve's information about the key either is impossible or considerably reduces the length of the key, so we leave this 1 bit information exposed to Eve. (This 1 bit is out of the final key, no matter how long the final key is.)

Combining the above steps yields the fixed scheme. To cope with the non-stable channel condition, in the online scheme, we dynamically adjust the reading threshold α in the level crossing algorithm by an online parameter learning scheme, which is presented in Section 6. Experiments show that adjusting parameters, even one parameter, can help tolerate non-stable channel conditions.

3.3 Performance Metric

To determine how well our scheme works, we define the metric *approximate entropy bit rate* as

$$a_{bps} = E \cdot (n_q - n_p)/t,$$

where E is the estimated Shannon entropy per bit of the quantized bits, n_q is the number of the quantized bits, n_p is the number of exchanged parity bits during reconciliation, and t is the time duration of the trace. Note that this metric has taken into account the leaked information during reconciliation.

We also define several related terms. First, *quantized bit rate*: $q_{bps} = n_q/t$. It is the raw rate of level crossing. It does not take into account the leaked information. Second, since we select a randomness extractor to actually extract the entropy, we have *secret bit rate*: $s_{bps} = (n_e - n_p \cdot n_e/n_q)/t$, where n_e is the number of extracted bits by our randomness extractor. It does take into account the leaked information. We have $s_{bps} \leq a_{bps} \leq q_{bps}$. The metric a_{bps} reflects the entropy bits per second. The reason for $s_{bps} \neq a_{bps}$ is that our implementation of the randomness extractor does not achieve an efficiency equal to Shannon entropy, but it already outperforms existing extraction schemes relying on random hashing. The details are explained in Section 5.2.

4 Smoothing

The measured RSSI in vehicular environments contains heavy background noise that severely affects the performance of any key establishment algorithm. We propose to use weighted sliding window smoothing to reduce the effect of noise. Considering that the measured RSSI at two parties should be the same in the case without noise, our smoothing technique aims to make the RSSI at each location as similar as possible. The similarity is quantified by correlation coefficient. In this section we introduce how to smooth the readings, and in Section 7.3 we compare our approach with existing methods.

The task is formulated as an optimization problem. For a fixed window size k (to be determined in experiments), we assign different weights $\mathbf{a} = (a_1, a_2, \dots, a_k)$ and $\mathbf{b} = (b_1, b_2, \dots, b_k)$ to the readings at Alice and Bob respectively. The weights should satisfy $\sum a_i = 1$ and $\sum b_i = 1$. Using these weights, we have the ith smoothed reading $x'_i = \sum_{j=1}^{k} a_j x_{i+j-1}$ for Alice and $y'_i = \sum_{j=1}^{k} b_j y_{i+j-1}$ for Bob. Let $X' = (x'_1, x'_2, \dots, x'_{n-k+1})$ and $Y' = (y'_1, y'_2, \dots, y'_{n-k+1})$. We will select the weights $\mathbf{a}, \mathbf{b}$ to maximize the correlation coefficient of the smoothed readings, i.e., we need to solve

$$\max_{\mathbf{a},\mathbf{b}} \quad \rho_{X',Y'}. \tag{1}$$

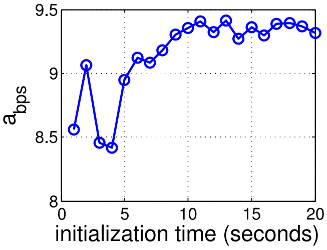


Fig. 5. Impact of initialization period on a_{bps} when smoothing window size k is 3.

This problem can be solved under a framework called *canonical correlation analysis (CCA)* [33]. Here we give a brief introduction to CCA. Given two matrices $A \in \mathbb{R}^{n \times d_1}$ and $B \in \mathbb{R}^{n \times d_2}$, CCA focuses on finding a linear combination of A's column vectors and a linear combination of B's column vectors such that the correlation coefficient of the two new vectors is maximized.

We can transform problem (1) into CCA as follows. Let $\mathbf{x}_i^j$ denote the column vector $(x_i, x_{i+1}, \dots, x_j)$ and let $\mathbf{y}_i^j$ be defined similarly. Let $A = \left[\mathbf{x}_1^{n-k+1}, \mathbf{x}_2^{n-k+2}, \dots, \mathbf{x}_k^n\right]$ and $B = \left[\mathbf{y}_1^{n-k+1}, \mathbf{y}_2^{n-k+2}, \dots, \mathbf{y}_k^n\right]$, where $A, B \in \mathbb{R}^{(n-k+1)\times k}$. Applying CCA on A and B yields two linear combinations. The linear combination for A gives the optimal $\mathbf{a}$, and the linear combination for B gives the optimal $\mathbf{b}$. Now we have the optimal solution to problem (1). The computation involves eigenvalue decomposition to a matrix with size $k \times k$. Given that k is usually very small in our case, it is computationally efficient to solve the problem. After solving problem (1) and learning the two weights, Alice and Bob smooth the rest of the data using the learned weights.

Note that the input to problem (1) includes RSSI data from both Alice and Bob, which is not available at either party unless communication. Therefore, to use this smoothing scheme, there should be an initialization period, during which Bob sends his observed RSSI data to Alice. These RSSI data sent are not encrypted and are disclosed to the adversary, so we do not use any RSSI data during this period for key extraction, and only leverage them to learn the smoothing weights, which reflect the noise level and sampling offset. Specifically, Alice solves problem (1) with both parties' data, and sends the optimal $\mathbf{b}$ to Bob. Then, Alice smooths her subsequent RSSI data by $\mathbf{a}$ and Bob smooths his data by $\mathbf{b}$ independently. The smoothed data are input to the level crossing algorithm. To determine the appropriate initialization period, we try different periods and run our algorithm on RSSI data from a real-world vehicular experiment. Fig. 5 shows that a 10-second initialization time is enough.

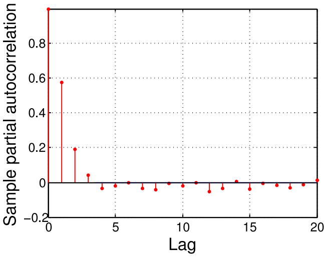
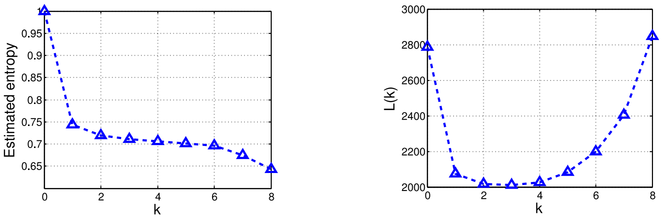


Fig. 6. Partial correlation coefficient for a bit sequence generated from a real-world data set.



(a) Estimated entropy

(b) L(k)

Fig. 7. Estimated entropy and likelihood for quantized bits.

One concern of our smoothing method is that, although the raw RSSI data are unknown to the adversary, the smoothed RSSI data may be a function of the distance between Alice and Bob, so the smoothed data may not be safe if the adversary knows this distance. This concern is closely related to the window size k of our smoothing method with larger window sizes resulting in a bigger concern. In our experiments, we use a very small window size ($k = 3$) because it achieves better performance, and we believe that the smoothed data obtained from only three consecutive RSSI readings cannot be predicted from distance. Note that level crossing, the next step following smoothing, will further subtract slow variation from the smoothed data. Here, the subtracted slow variation is believed to be caused by changes in distance [4].

5 Extracting a Perfect Random Key

The data obtained by applying our smoothing method to the raw RSSI data are input to the level crossing algorithm, which generates quantized bits for both Alice and Bob. Some of Alice's bits may be different from that of Bob's due to noise. These bits are called mismatches, and are corrected by information reconciliation techniques. Now Alice and Bob have the same quantized bits. In this section, we describe a method for Alice such that she can extract a perfect random key from her quantized bits. Here perfect random means that the bits of the key should be independent from each other and the probability of a bit being 1 should be 1/2. Bob will also use this method and get the same final key since Alice and Bob have the same quantized bits. In the following, we use a sequence of the quantized bits obtained by applying level crossing to a real-world trace as an example to study the dependency issue and extract a perfect random key.

5.1 Dependency Modeling

We first show that quantized bits have limited dependency (each bit depends on finitely many previous bits), then we use a Markov chain to model the dependency.

To show the limited dependency property, we plot in Fig. 6 the partial autocorrelation coefficient (pacf) [34] of the quantized bits of one trace. Briefly, $pacf$ describes the correlation between two bits after eliminating the influence of bits in-between, which is different from autocorrelation, which does not eliminate the influence of bits in-between. We can see from the figure that for $lag \geq 4$, the absolute value of $pacf$ is very small, meaning each bit only depends on the previous 3 bits. This observation shows that it is appropriate to model the dependency using a Markov chain.

In order to do this, we need a method to determine the chain's "order", i.e., the number of previous bits a bit depends on. For example, if each bit only depends on one previous bit, then we can use a 1st order Markov model with state space $S^{(1)} = \{0, 1\}$. If each bit depends on two previous bits, then we need to use a 2nd order Markov model with sate space $S^{(2)} = \{00, 01, 10, 11\}$. Generally, we denote the sate space of a kth order Markov model by $S^{(k)}$. We need to estimate k.

The order is estimated using the popular BIC (Bayesian Information Criterion) Markov order estimator [35]. We first introduce how it works, and then show that it is appropriate in our case. Let n be the total number of bits. Let $\mathbf{s}_1^j = (s_1, s_2, \ldots, s_j)$ and $N(\mathbf{s}_1^j)$ be the number of occurrences of substring $\mathbf{s}_1^j$ in the bit string. Then the estimated order k is the number that minimizes the objective function $L(k)$

$$\min_{k} \quad L(k) = -\log P_{ML(k)} + 2^{k-1} \log n, \tag{2}$$

where $P_{ML(k)}$ is the kth order maximum likelihood with $\log P_{ML(k)} = \sum_{\mathbf{s}_1^{k+1} \in S^{(k+1)}} N(\mathbf{s}_1^{k+1}) \log \hat{p}(\mathbf{s}_1^{k+1} \,|\, \mathbf{s}_1^k)$ where $\hat{p}(\mathbf{s}_1^{k+1} \,|\, \mathbf{s}_1^k)$ is the empirical conditional probability of string $\mathbf{s}_1^{k+1}$ given $\mathbf{s}_1^k$. We find from experiments that the order of our quantized bits is always between 1 and 10. Thus we can solve Eq. (2) by enumerating $L(k)$ and picking the optimal k. In the situation where the order is beyond this range, we can also use iterated grid search described in Section 6.

After the order is estimated, the model is fully established and we can use the model to estimate the entropy of the quantized bits, which is the upper bound for the number of extracted bits [27], and is used in calculating leaked information during reconciliation. By definition, for a kth order Markov chain, the entropy rate is $H = -\sum_{i \in S^{(k)}} \pi(i) \sum_{j \in S^{(k)}} p(i,j) \log p(i,j)$ where $\pi(\cdot)$ is the stationary distribution and $p(i,j)$ is the transition probability from state i to j.

The chosen BIC Markov order estimator is appropriate in the sense that its output is consistent with two other heuristic approaches. The first heuristic approach is to compute $pacf$ for different lags, as in Fig. 6, and output the lag at which $pacf$ is very small. The second heuristic is to repeatedly increase the order of the Markov chain by one and output the order where the estimated entropy converges. For the quantized bits in Fig. 6, we have seen that the first heuristic approach estimates the order as 3. The second heuristic approach gives the same order. As shown in Fig. 7a, the estimated entropy is roughly the same around $k = 2, 3, 4$. Note that the concave part ($k > 4$) in this figure is caused by sample size limitation. Now consider the BIC estimator.

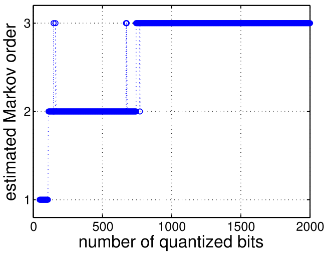


Fig. 8. Convergence of estimated markov order with respect to the number of quantized bits.

Fig. 7b depicts its objective L for the same quantized bits, indicating that the BIC estimator gives the same order estimation. The reason we do not use the two heuristic approaches is that we are unable to quantify “very small” and “convergence” in such a way that the resulting estimator has provable properties such as consistency.

In practice, another issue for Alice (Bob) is to determine how many quantized bits are needed for estimating the Markov order. Clearly, a small sample size results in estimation inaccuracy, while a large sample size requires longer probing time. To this end, for the quantized bits studied in Fig. 6, we apply the BIC estimator to different numbers of the quantized bits, and show the estimated Markov order in Fig. 8. We can see that the estimated order converges to 3 when the number of the quantized bits exceeds 800. This convergency is expected since the BIC estimator has been proved to be consistent [36]. Therefore, in practice, Alice (Bob) keeps re-estimating the Markov order whenever new quantized bits are produced by level crossing, until the estimated Markov order converges (e.g., it has not changed in the latest 1000 quantized bits). Note that the quantized bits used for Markov order estimation are not disclosed to Eve.

5.2 Randomness Extractor

We first consider extracting perfect random bits from the dependent quantized bits. It is not safe to simply perform random hashing based on our estimated entropy, which is Shannon entropy, because random hashing should be based on *min-entropy* [26]. For extracting perfect random bits from a Markov chain, there is a simple and elegant method [37], whose efficiency, unfortunately, is low and cannot be improved to approach Shannon entropy. Thus we turn to an earlier approach [27], which is more complex but the efficiency can approach Shannon entropy.

The randomness extractor consists of two steps. First, the bits are split into subsequences such that each subsequence contains independent bits. A bit belongs to the subsequence corresponding to the state determined by the bit's previous k bits, where k is the estimated order of the Markov model. Thus, each subsequence corresponds to a distinct Markov state, resulting in 2^k subsequences in total. The splitting process can be done efficiently by scanning the bits from left to right. Fig. 9 gives an example of splitting bits from a 2nd order Markov chain into $2^2 = 4$ subsequences, S_{00}, S_{01}, S_{10} and S_{11}. A dash box starts from the leftmost two bits and slides toward the right side one bit per step. In each step, the two bits inside the dash box show one of the four subsequences, indicating that the right-hand bit, outside the dash box, belongs to that subsequence. This procedure repeats until the dash box reaches the end of the sequence.

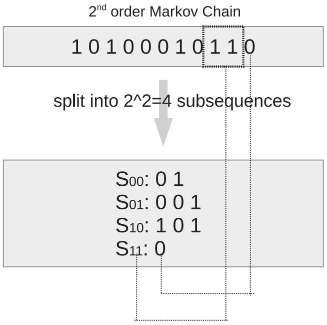


Fig. 9. Independent subsequence splitting. For instance, bits 11 in the dash box indicates that the current state is 11 so that the next bit, 0, belongs to the subsequence S_{11}.

It can be proven that each resulting subsequence of the splitting procedure contains independent bits.

Second, we extract *unbiased* bits from each subsequence, and merge these bits into a single bit stream. A bit is *unbiased* if the probability of it being 1 is equal to $1/2$, otherwise it is biased. Each subsequence is divided into blocks of N bits. N is an important parameter influencing the extractor's efficiency, i.e., the ratio of output bits to input bits. For any N-bit block, if there are i ones, then all the $\binom{N}{i}$ possible cases happen with equal probability. Thus, we can encode each of these $\binom{N}{i}$ cases with $\log \binom{N}{i}$ bits on average. In practice, we construct a table encoding every batch of N biased input bits into a variable length code by considering all possible is. One such encoding table for $N = 4$ is given in Table 1, where i is the number of ones and $\wedge$ means null. For example, this table encodes 0010 into 01, 1100 into 11, and 0011 into 0. So, an independent biased bit sequence 0010 1100 0011 will be encoded by this table into an unbiased bit sequence 01 11 0 with an efficiency of 0.417. Note that the encoding table can be constructed off-line and there are many tables for a given N. It does not matter which table is used, as long as they use the same table.

The block size N influences the efficiency, i.e., the ratio between output bits and input bits, of the resulting extractor. It is known that when N approaches infinity, the efficiency of the adopted randomness extractor approaches Shannon entropy [27]. Fig. 10 shows our simulation results. We can see that, for a fixed N, the efficiency varies with respect to p, and for a fixed p, it decreases with the increase of block size N. Due to inability to control p, we prefer larger block sizes for improved efficiency. Unfortunately, larger block sizes require a larger encoding table, on the order of $O(2^N)$. Consequently, the block size N in our current implementation is 20, and we plan to further increase N in the future for better efficiency (in other words, long

TABLE 1
A Possible Encoding Table For $N = 4$

i	0	1				2						3				4
seq	0 0 0 0	0 0 0 1	0 0 1 0	0 1 0 0	1 0 0 0	0 0 1 1	0 1 0 1	0 1 1 0	1 0 0 1	1 0 1 0	1 1 0 0	1 1 1 0	1 1 0 1	1 0 1 1	0 1 1 1	1 1 1 1
code	$\wedge$	0 0	0 1	1 0	1 1	0	1	0 0	0 1	1 0	1 1	0 0	0 1	1 0	1 1	$\wedge$

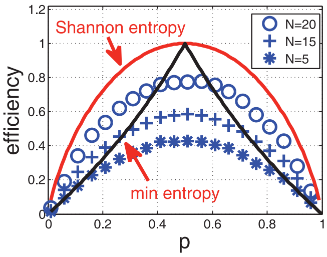


Fig. 10. Efficiency of the randomness extractor. For each p = 0.05, 0.10,0.15,...,1.00, we simulate 4,000 independent biased random bits, and extract unbiased random bits using an encoding table with block size N. Note that, in theory, the efficiency approaches Shannon entropy as N approaches infinity.

keys). It is worth mentioning that an extractor with $N = 20$ can already outperform existing extraction schemes relying on random hashing mentioned in Section 2, whose efficiency is upper bounded by the min entropy and is generally smaller than our extractor as in Fig. 10. The limitation of finite N is reflected on the difference between s_{bps} and a_{bps}, which we evaluate in Section 7.5.

5.3 Model Discussion

In this section, we consider the necessity and validity of our Markov model, and study the impact of the Markov order estimation inaccuracy on the final extracted key.

One may suggest to pick every kth quantized bit, because each quantized bit depends on the previous $k-1$ quantized bits. The problem is that the chosen bits may still be dependent. For example, suppose we have four bits, $abcd$, where each bit depends on the previous bit. Picking every 2nd quantized bit means that we pick b and d. However, we can easily prove that the probability of bit d being 1 is dependent on bit b. Generally, picking every kth quantized bit yields a dependent bit sequence, which renders the extracted key not perfectly random, and the key cannot pass the NIST test. To this end, we believe that the Markov model is necessary to capture the dependency among the quantized bits, so that this dependency can be eliminated.

Another concern of our Markov model is how well it approximates the ground truth. However, the ground truth is unknown. To prove the validity of our Markov model, we have observed the finite dependency property and provide empirical evidence, which are in line with what is expected if the Markov model approximates the ground truth well. First, each subsequence produced by the first step of our randomness extractor contains independent bits, so the bits should be uncorrelated. This is confirmed by Fig. 11, which plots the autocorrelation of one subsequence from our experiments. Second, the bits extracted by the randomness extractor are perfectly random, so they should be able to pass the NIST test. As shown in our experiments, all of our extracted bits do pass the test. (Without our extractor, none of them can pass the test.)

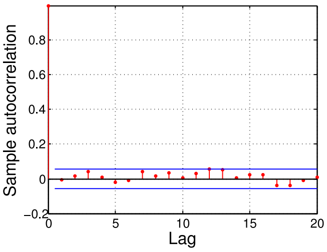


Fig. 11. Autocorrelation of a subsequence. Bits are uncorrelated within 95 percent confidence interval.

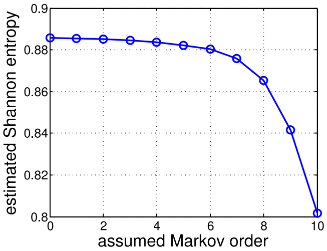


Fig. 12. Impact of overestimation of Markov order on the estimated per-bit Shannon entropy.

Consider the case where Markov model is appropriate, but the estimated Markov order may be larger (overestimation) or smaller (underestimation) than the ground truth. As mentioned before, any Markov chain contains smaller order Markov chains as special cases. Thus, in the case of overestimation, the extracted bits are still perfectly random and can pass the NIST test. The drawback is that the estimated a_{bps} may be smaller than the ground-truth, because overestimation in the Markov order results in underestimation of the per-bit Shannon entropy due to sample size limitation. We show this issue with numerical simulation, where the ground truth model is a 0-th order Markov chain (i.e., bits are independent), and we generate 10,000 bits with a bit being set to 1 with probability 0.3. We estimate the entropy of the generated bits by an ith order Markov chain for $i = 0, 1, \ldots, 10$. Fig. 12 shows that the estimated Shannon entropy decreases with the increase of assumed Markov order. In contrast to overestimation of the Markov order, if the estimated Markov order is smaller than the ground truth, then the extracted bits can no longer be guaranteed to be perfectly random and pass the NIST test. This is because dependency has not been eliminated from each subsequence. Therefore, to guarantee perfect randomness of the extracted key, we are allowed to overestimate the order of the Markov chain but *not* underestimate it.

6 Online Parameter Learning

Channel dynamics of vehicular networks are not stable, thus a fixed set of parameters may not work well in all situations. To address this issue, we consider how to adjust some parameters in an online fashion. In this section we propose an online scheme to learn a suitable quantization threshold α dynamically. The reason for adjusting α instead of m and s is discussed in Section 6.3.

In this online scheme, suppose Alice is the leader. For a training period, Bob sends his residual readings to Alice. Alice simulates the key generation process on her own readings and Bob's readings with different α. Then she picks the α that optimizes the resulting bit rate, and sends it to Bob. After training period, both sides will use this α for key generation. But how to optimize α and which training period should be used?

6.1 Choosing α

One natural solution is to try several α values, and pick the one that maximizes a_{bps}. Due to the intermediate steps, such

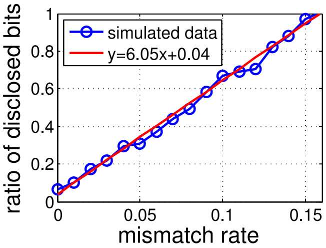


Fig. 13. Impact of mismatch rate on the amount of disclosed bits during information reconciliation.

as entropy estimation and randomness extraction, this method is computationally intensive. Instead, we give the following function f as a qualitative approximation of a_{bps}. For a given α, the leader Alice, simulating level crossing with α, obtains the corresponding quantized bit rate q_{bps} and its mismatch rate γ_{mis}. The function f translates each (q_{bps}, γ_{mis}) pair to a scalar:

$$f(q_{bps}, \gamma_{mis}) = q_{bps}(0.96 - 6.05\gamma_{mis}).$$

This function can be computed quickly for a certain α without performing entropy estimation and information reconciliation, and is designed based on the following empirical study. We simulate two bit sequences of 4,000 bits with different mismatch rates, and apply the implemented information reconciliation method to correct mismatches. We study the relationship between the ratio of disclosed bits to total bits and the bits' mismatch rate. Fig. 13 shows that the ratio of disclosed bits is approximately a linear function of the mismatch rate in the form of $y = 6.05x + 0.04$, which means that for two bit sequences with a mismatch rate x, a portion of $6.05x + 0.04$ bits will be disclosed during information reconciliation. The heuristic metric, function f, follows immediately. This metric is heuristic in that it did not take entropy into account. Because the linear function $y = 6.05x + 0.04$ is closely related to the implementation of the information reconciliation method, it is worth mentioning our implementation of the Cascade protocol [29]. We use three iterations of reconciliation, and due to the inability to determine the mismatch rate beforehand, we use, as suggested by Jana et al.[6], a random block size. This block size is a random number in the range from 16 to 256 in our implementation. Note that in other situations where the mismatch rate is known or can be estimated accurately, using the optimal block size, instead of a random block size, discloses less information [38], and a different heuristic metric can be used.

Fig. 14 shows the two metrics f and a_{bps} for bits generated in a vehicular experiment, where negative f is rounded to 0 for clarity. The level crossing algorithm generating the bits has a window size (s) of 1,000 and an excursion threshold (m) of 2. We can see that the optimal solution to f is very close to a_{bps}, indicating that we only need to solve the optimization problem with objective f. To this end, we observe a property of f that expedites the exhaustive search for solving the optimization problem: f is approximately unimodal. Considering this property, we use the iterated grid search algorithm [30]. (Refer to the conference version [10] for the algorithm.)

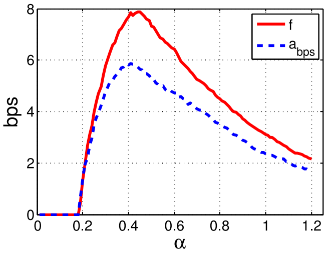


Fig. 14. f and a_{bps} with respect to α on one dataset. Note that negative f is rounded to zero.

6.2 Training Period

Choosing different training periods may give quite different performance. We consider two schemes in this section. The first is a naive scheme called single-frame. Time is divided into frames, each of which has length l measured by the number of readings. In each frame, the leader uses the first $l \cdot \gamma_r$ ($\gamma_r < 1$) readings for training, obtains the best α, and then applies it to the remaining readings of the frame. We find from our experiments that this scheme does not work well in terms of a_{bps} due to the fact that the channel condition is not stable. Specifically, some frames have poorly estimated α causing many mismatched bits in that frame. These mismatched bits significantly reduce the overall a_{bps}, since each mismatched bit causes the disclosure of more than 5 bits during information reconciliation, which is indicated in Fig. 13. It is impossible to identify these bad frames by Alice or Bob without cooperation.

Observe that in the single-frame scheme, each non-training period is between two training periods from adjacent frames. For example, let T denote training period and N denote non-training period. Then the single-frame scheme is $TNTNTN\ldots$, where each N is between two Ts. We propose a double-frame scheme. Previously in the single frame scheme, the α applied to each non-training period is based on the single frame the period belongs to. In the double-frame scheme, the α is based on the two adjacent frames. It is the bigger of the two estimations, made in the two frames. We chose the bigger of the two, because the metric a_{bps} is not symmetric with respect to the optimal α in that, for a given distance to the peak, a smaller α reduces a_{bps} more significantly than a larger α, which is shown in Fig. 14.

6.3 Discussion

The message exchanging described in this section does not compromise the security level. First, it is almost impossible to predict the fluctuation of the residual readings from exposed historical data segments nearby, due to the unpredictable nature of wireless dynamics. Thus, the adversary cannot learn more about the final secret bits generated from the unrevealed residual data segments. Second, disclosed parameters after training only reflect the statistics of the noise. This information may indicate the secret bit rate of our scheme, but this is insignificant since the secret bit rate is publicly known.

Our online scheme adjusts α in level crossing instead of the other two parameters for two reasons. First, the influence of the three parameters α, m and s on a_{bps} are not independent in that the inefficiency caused by one parameter can be, to some extend, compensated by the other parameters. For example, increasing any one of the parameters can reduce

TABLE 2
Dataset Descriptions (There Are Two Routes, Suburban Route (sub.) and Rural Route (rur.))

no.	1	2	3	4	5	6	7	8
rate	500	500	500	500	500	1000	1000	1000
route	sub.	sub.	sub.	sub.	sub.	rur.	rur.	rur.
speed	25	35	35	45	45	35	45	50
dura.	472	455	432	338	319	234	219	227
corr.	0.39	0.48	0.49	0.52	0.49	0.26	0.31	0.33

The Units for Probing Rate (Rate), Driving Speed (Speed), and Trace Duration (Dura.) are probes per second, mPh, and seconds, respectively. Correlation coefficient (corr.) is calculated based on residual data mentioned in Section 3.1, which is obtained by subtracting a windowed moving average from the raw data.

the mismatch rate of the quantized bits, so a higher mismatch rate caused by a smaller s can be compensated by increasing α. Second, during our experiments we find that a_{bps} is very sensitive to α, but insensitive to s. We also find that setting $m = 2$ consistently outperforms other settings.

7 Evaluation

For our evaluation, we first collected trace data using experiments performed on actual vehicles in a realistic setting. We then study the impact of parameters and compare our smoothing methods with existing ones based on one dataset, and then apply our method to all datasets to show how fast we can extract bits. Finally we validate the randomness of the final bits by the NIST test suite. Note that the two objectives a_{bps} and s_{bps} are in terms of "bits per second", and are the average bit rate with respect to each data trace.

We do not quantitatively compare our scheme against [4], [5], [6], [7], because we feel it might not be completely fair since they do not encounter the strong noise in our scenario. Our work builds upon level-crossing [4], [5], but the default parameter settings either lead to a key generation rate of 0 bit per second, or a key that cannot pass the NIST test in our experiments. We run ASBG [6] on one vehicular trace and find a mismatch rate over 44 percent, using single-bit scheme that has lower mismatch rate than multi-bit extraction scheme. Consequently, the resulting bit rate is rather low, nearly 0. Reference [7] describes a smoothing method for reducing noise, a de-correlation method for removing dependency, and a multi-bit quantization method for faster key generation. We compare the smoothing method in Section 7.3. Since the de-correlation method has security concern (a de-correlated bit sequence may still be dependent as mentioned in Section 2), we do not include it for comparison. The multi-bit quantization method has the lowest mismatch rate when only one bit is generated for a sample. In this case, it is essentially equivalent to ASBG, and has a mismatch rate over 40 percent.

7.1 Experimental Setup

We used two vehicles, Alice and Bob, for our experiments. Each vehicle is equipped with a laptop running Ubuntu 9.10 with MadWifi 0.94 wireless driver for Atheros chipsets and external antennas mounted on top of the car. WiFi transmit power is set to the default value.

We selected two different testing routes to represent a suburban environment and a rural environment (Refer to the conference version [10] for a GoogleEarth picture). We performed a total of 8 runs over the two routes to arrive at 8 datasets total. Table 2 summarizes the 8 datasets. Datasets $1 \sim 5$ are from the suburban route with rate of 500 probes per second, while datasets $6 \sim 8$ are from the rural route with 1,000 probes per second. It is worth mentioning that these probing rates are for the establishment of the secret key, which takes only a few seconds.[2]

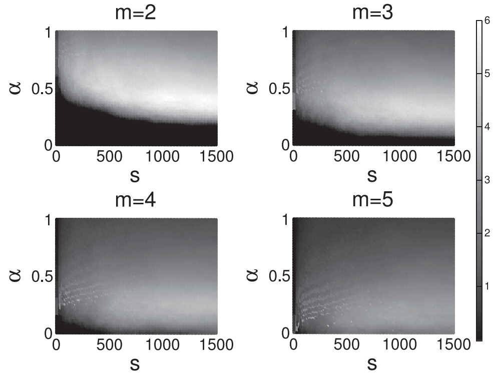


Fig. 15. Influence of three parameters on objective a_{bps}.

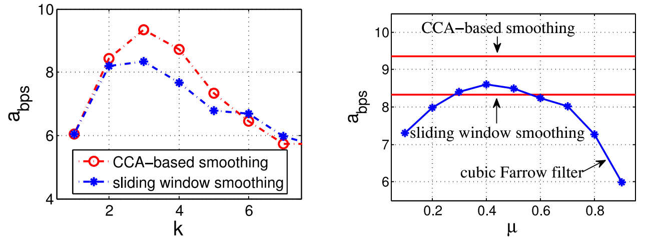
For every run, we drove the vehicles in a single-file manner, one in front of the other, maintaining a distance of 5 to 10 meters between the two vehicles. During a single run of the route, the car Alice will continuously send probe messages to Bob. These probe messages will only have a preamble and an index payload. Bob will immediately return the corresponding replies upon reception of this message. Both Alice and Bob will record, as datasets, the RSSI readings of received packets, along with their indices.

7.2 Impact of Level Crossing's Parameters

In our approach, level crossing is an important step that involves three parameters: sliding window size s, thresholds α and m. To study the impact of these parameters on the final a_{bps}, we apply our approach (without smoothing step) to a vehicular trace data with different parameter settings of level crossing. We vary s from 10 to $1,500$ with increments of 10, α from 0.00 to 1.00 with increments of 0.01 and m from 2 to 5 with increments of 1. For each parameter setting, we compute the final a_{bps}, and show the result in Fig. 15.

We can see from Fig. 15 that different parameter settings greatly affect the achieved a_{bps}. When both m and s are fixed, the objective first increases with respect to α, and, after a turning point, starts to decrease, so the turning point maximizes a_{bps}. However, the turning point for different m and s is different. Generally, the turning point decreases with respect to s when m is fixed, and also decreases with respect to m when s is fixed. This is because, increasing either s or m can already reduce the mismatch rate of the quantized bits, and in this case smaller α helps to capture more signal variations. We can observe that reasonably well a_{bps} for this dataset can be obtained when $s = 1,000$, $m = 2$, and $\alpha = 0.4$. This will be the benchmark parameter setting

[2] During this period, the two parties cannot transmit normal messages due to lack of a secret key, so a high probing rate does not necessarily incur much extra overhead compared to a low probing rate. A high probing rate can speed up key generation by capturing more channel variations.



(a) Sliding window smoothing and CCA-based smoothing.

(b) Impact of parameter μ on cubic Farrow filter.

Fig. 16. Comparing the three smoothing techniques, sliding window smoothing, CCA-based smoothing, and cubic Farrow filter in vehicular environment. No smoothing is performed when $k = 1$. The window size for both sliding window smoothing and our approach is 3 in (b).

in the experiments. To deal with different probing rates, we generalize window size s as the number of readings in 2 seconds, since the probing rate of this dataset is $500/\mathrm{s}$.

7.3 Comparing Smoothing Methods

We compare our smoothing method with two other methods used in the key extraction literature, which differ from ours in how the smoothing weights $\mathbf{a}, \mathbf{b}$ are selected. The first is the well-known sliding window smoothing used in [14]. Both $\mathbf{a}$ and $\mathbf{b}$ in this method are fixed as $(1/k, 1/k, \ldots, 1/k)$. The second is a cubic Farrow filter [39] used in [7], [19]. The purpose of this filter is to interpolate the sensed RSSI of Alice and Bob at some common time instant, since Alice and Bob cannot sense the channel at the same time due to half-duplex wireless transceivers they have. The window size k of this approach is 4, and two parameters, the fractional delay μ_A for Alice and μ_B for Bob, determine the weights $\mathbf{a}$ and $\mathbf{b}$ in the following manner [39][3]: $\mathbf{a} = \big[-\frac{\mu_A^3}{6} + \frac{\mu_A^2}{2} - \frac{\mu_A}{3},\ \frac{\mu_A^3}{2} - \mu_A^2 - \frac{\mu_A}{2} + 1, -\frac{\mu_A^3}{2} + \frac{\mu_A^2}{2} + \mu_A, \frac{\mu_A^3}{6} - \frac{\mu_A}{6} \big]$, and $\mathbf{b} = \big[-\frac{\mu_B^3}{6} + \frac{\mu_B^2}{2} - \frac{\mu_B}{3},\ \frac{\mu_B^3}{2} - \mu_B^2 - \frac{\mu_B}{2} + 1, -\frac{\mu_B^3}{2} + \frac{\mu_B^2}{2} + \mu_B, \frac{\mu_B^3}{6} - \frac{\mu_B}{6} \big]$. For μ_A and μ_B, let τ_B be the time when Bob receives a probe message, τ_A be the time when Alice receives the corresponding ACK in a probe-ACK round, and T be the time duration between two probe messages. Then $\mu = \frac{1}{2}\big[\frac{\tau_A - \tau_B}{T}\big]$ determines μ_A and μ_B. Reference [7] sets $\mu_A = 1 - \mu$ and $\mu_B = 1 + \mu$.

We implement the three smoothing methods and compare them on a vehicular trace data. For each method, its smoothed data are fed into level crossing, whose parameters are set as $s = 1000, m = 2$ and $\alpha = 0.4$, and we measure and compare the final a_{bps}. The smoothing method we proposed is referred to as CCA-based smoothing, and it uses a 10-second initialization time for obtaining the optimal $\mathbf{a}$ and $\mathbf{b}$. First, we compare our approach with sliding window smoothing under different window sizes. Fig. 16a shows that our approach generally outperforms sliding window smoothing, and both of them achieve the highest a_{bps} when the window size is 3. The performance decreases for $k > 3$ because some useful variations are smoothed out. Second, we compare the two approaches with the cubic Farrow filter. Due to lack of time synchronization during the experiment, we are not able to determine μ of the cubic Farrow filter, thus we study this filter by varying μ from 0.1 to 0.9 with increments of 0.1. It is worth mentioning that directly plugging $\mu_B = 1 + \mu$ into $\mathbf{b}$ results in a performance worse than that of sliding window smoothing. Instead, we plug $\mu_B = \mu$ into $\mathbf{b}$, and delay the sample index of Bob by one, according to [7]. Fig. 16 shows that the highest a_{bps} of cubic Farrow filter is between that of sliding window smoothing and that of our approach. In summary, our approach performs better than existing solutions, and it generates $50\,\mathrm{percent}$ more entropy bits than the case without smoothing. We believe the benefit of our approach comes from both reduced local noise and adjusted sensing time.

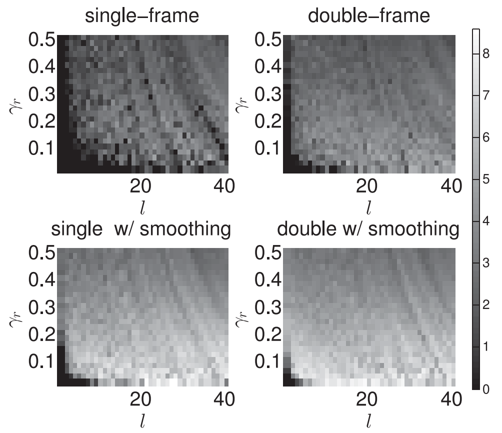


Fig. 17. Performance of single-frame scheme and double-frame scheme on non-smoothed data (the two top figures) and smoothed data (the two bottom figures). Metric a_{bps} is represented by color, and frame length l is in units of seconds.

7.4 Comparing Online Schemes

We compare single-frame scheme and double-frame schemes under two cases. In the first case, the comparison is done using the raw RSSI data, and in the second case, the raw RSSI data are preprocessed by our smoothing method with weights learnt during a 10-second initialization period. In each case, we vary $\gamma_r = 0.02, 0.04, \ldots, 0.50$ and $l = 1, 2, \ldots, 20$, where l is in units of seconds, i.e., $l = 1$ indicates that each frame consists of the readings in 1 second. The results are shown in Fig. 17.

We can see from the figure that our smoothing method, though spending 10 seconds for learning smoothing weights, greatly increases a_{bps} for both single-frame scheme and double-frame scheme. Additionally, double-frame scheme significantly outperforms single-frame scheme in the case without smoothing. In the case with smoothing, it is still better than single-frame scheme in the sense that there are more (γ_r, l) pairs leading to very good performance. For the double-frame scheme with smoothing, we observe that $\gamma_r = 0.02 \sim 0.10$ and $l = 10 \sim 20$ perform equally well so that we set $\gamma_r = 0.05$ and $l = 20$ in other experiments if they are not specified.

7.5 Comparing Parameter-Selection Schemes

Based on previous studies, we compute the approximate entropy bit rate for all datasets. We compare three schemes according to how α is chosen. (Two of them are mentioned in Section 3.2.) (1) Oracle-assisted scheme (oracle). In this scheme, the α for a dataset is chosen by an oracle using the

[3] The formula in [7] missed "+1" in a_2 (the 2nd element in $\mathbf{a}$) and b_2.

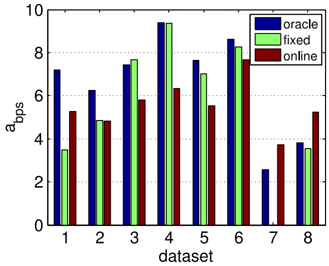


Fig. 18. Approximate bit rate of all datasets.

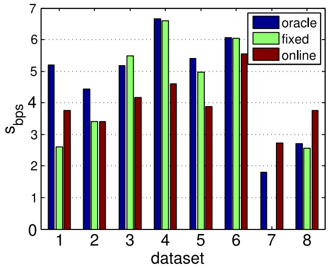


Fig. 19. Secret bit rate of all datasets.

iterative grid search algorithm. The oracle is assumed to know both Alice and Bob's data, so this scheme is impractical and is only for comparison. (2) Fixed scheme (fixed). The α in this scheme is fixed as 0.4 for all datasets. (3) Online scheme (online). This is the double-frame scheme described in Section 6. All three schemes are applied to smoothed data obtained by CCA-based smoothing with smoothing window size 3 and smoothing weights learned from the first 10-second RSSI readings of the corresponding dataset. We show in Fig. 18 the resulting approximate bit rates and in Fig. 19 the resulting secret bit rates of all datasets. We have the following observations.

First, there is no obvious correlation between driving speed (shown in Table 2) and a_{bps} for the oracle-assisted scheme, so the main factor limiting a_{bps} is the noise contained in the readings rather than insufficient channel variations as before [4], [6]. Second, the fixed scheme generally performs worse than the oracle-assisted scheme. This coincides with intuition since the oracle-assisted scheme assumes the knowledge of both Alice and Bob's RSSI data. In dataset 3, the fixed scheme performs slightly better than the oracle-assisted scheme, because the algorithm we use in the oracle-assisted scheme can only approximately, instead of accurately, find the optimal α as mentioned in Section 6. The a_{bps} of the fixed scheme is 0 for dataset 7. Exhaustive search over α on this dataset shows that α should be over 0.42 to achieve non-zero a_{bps}, but the fixed scheme sets it as 0.4. This observation shows the necessity of our online scheme that can adjust α in different environments. Third, the online scheme offers more consistent performance than both the oracle-assisted scheme and the fixed scheme. It can get 4-8 entropy bits per second for all datasets, and it performs better than the oracle-assisted scheme on datasets 7 and 8. In these two datasets, a single α for a whole dataset, though it is the best one in terms of a_{bps}, performs worse than several α even if these α are not the best ones for the corresponding time periods (frames). Due to the adjustment in each time frame, the online scheme offers more consistent performance in different environments. Fourth, by comparing Fig. 18 with Fig. 19, we can see that s_{bps} is smaller than a_{bps} due to the efficiency of the randomness extractor. As mentioned in Section 5.2, in principle, we can get s_{bps} arbitrarily close to a_{bps} by increasing N in the extractor. But there are computational challenges, so we reserve this for future work.

7.6 Randomness Validation

Recall that we consider passive attacks in this paper. In vehicular environments, the adversary, Eve, is at least a few meters away from both Alice and Bob. It has been theoretically proved (e.g., in [40]) that a third party, who is $\lambda/2$ away from Alice and Bob, experiences statistically independent fading channels (i.e., RSSI value variations) to Alice and to Bob, compared with the fading channel between Alice and Bob. Therefore, Eve cannot get any information about the dynamics experienced by Alice and Bob. In addition, the information exposed by information reconciliation has been removed by privacy amplification. Thus, the only security concern left is whether the extracted bits are sufficiently random. To validate randomness, for each s_{bps} in Fig. 19, we perform NIST tests on the extracted bits (before privacy amplification). Note that even in case where $s_{bps} = 0$ (fixed scheme on dataset 7 in Fig. 19), we have bits to be tested since the zero bit rate is caused by subtraction of leaked information. Among the 15 tests in the NIST tool, we run 8 of them, since the other tests require large input size and are not applicable to our case. We find that the extracted bits can pass all tests we conduct, ensuring the randomness of the final key. Table 3 shows the test results for oracle-assisted scheme. The results for the other two schemes are similar.

TABLE 3
NIST Test Results (p-Value) on Extracted Bits (Note that a p-Value Greater than 0.01 Indicates that the Corresponding Test is Passed)

Dataset	1	2	3	4	5	6	7	8
Frequency (Monobit)	0.54	0.52	0.66	0.40	0.65	0.54	0.77	0.31
Frequency within Block	0.76	0.15	0.70	0.50	0.32	0.38	0.55	0.16
Runs	0.88	0.77	0.94	0.67	0.80	0.39	0.28	0.54
Longest Run of Ones	0.76	0.75	0.57	0.48	0.83	0.86	0.41	0.94
Discrete Fourier Transform	0.27	0.15	0.32	0.56	0.03	0.72	0.64	0.53
Serial	0.69,0.73	0.71,0.27	0.64,0.93	0.47,0.48	0.69,0.85	0.59,0.27	0.74,0.42	0.87,0.94
Approximate Entropy	0.38	0.89	0.83	0.04	0.73	0.51	0.66	0.16
Cumulative sums	0.88,0.36	0.44,0.52	0.46,0.84	0.58,0.67	0.72,0.35	0.59,0.59	0.50,0.76	0.50,0.36

7.7 Discussion on Faster Key Generation

In this work, we focus on the single-channel scenario, i.e., we use RSSI measurements from a single channel. The resulting secret bit rate is around 5 bits per second. For applications that require faster key generation, there are several alternatives. The first is to use high precision measurement such as CIR [4] instead of RSSI. The methodology that we developed in this work is also applicable to the CIR measurements. The second approach is to explore the signal variations in the frequency domain. Signal variations at different channels are different from each other, which has been demonstrated in sensor motes [16]. For WiFi networks, the emerging multi-radio, multi-channel technique provides the opportunity for simultaneous transmissions on multiple channels. We can apply our approach simultaneously on all available channels, generating secret key multiple times faster than the single channel scenario.

8 Conclusions and Future Work

We consider extracting a shared secret key for two vehicles communicating over a wireless channel. Our solution extends an existing level crossing technique to work in noisy vehicular environments. Measurements from real world vehicular networks suggest that we can extract around 5 bits per second in most cases, and adjusting parameters in real-time can help tolerate noise in different environments and offer steady performance.

The key extraction speed of our scheme might not be sufficient for vehicles with too short contact time, e.g., when they are driving fast in the opposite direction. Adapting our scheme to such difficult situations is a promising research direction. In addition, it is also an important future work to defend against active attacks, i.e., attacks where the adversary could jam the transmission and influence the wireless channel.

Acknowledgments

This project was supported in part by China 973 project (2014CB340303), US National Science Foundation grants CNS-1320453, CNS-1117412, CNS-1156574, CNS-1449860, CNS-1527346, the National Natural Science Foundation of China (61502232, 61472252, 61321491), and China Postdoctoral Science Foundation (2015M570445). F. Xu is the corresponding author of this paper.

References

[1] W. Viriyasitavat, F. Bai, and O. K. Tonguz, “Dynamics of network connectivity in urban vehicular networks,” *IEEE J. Select. Areas Commun.*, vol. 29, no. 3, pp. 515–533, Mar. 2011.

[2] J. J. Haas, Y.-C. Hu, and K. P. Laberteaux, “The impact of key assignment on VANET privacy,” *Security Commun. Netw.*, vol. 3, no. 2/3, pp. 233–249, 2010.

[3] P. Papadimitratos, L. Buttyan, T. Holczer, E. Schoch, J. Freudiger, M. Raya, Z. Ma, F. Kargl, A. Kung, and J.-P. Hubaux, “Secure vehicular communication systems: Design and architecture,” *IEEE Commun. Mag.*, vol. 46, no. 11, pp. 100–109, Nov. 2008.

[4] S. Mathur, W. Trappe, N. Mandayam, C. Ye, and A. Reznik, “Radio-telepathy: Extracting a secret key from an unauthenticated wireless channel,” in *Proc. 14th ACM Int. Conf. Mobile Comput. Netw.*, 2008, pp. 128–139.

[5] C. Ye, S. Mathur, A. Reznik, Y. Shah, W. Trappe, and N. B. Mandayam, “Information-theoretically secret key generation for fading wireless channels,” *IEEE Trans. Inform. Forensics Security*, vol. 5, no. 2, pp. 240–254, Jun. 2010.

[6] 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 *Proc. 15th Annu. Int. Conf. Mobile Comput. Netw.*, 2009, pp. 321–332.

[7] N. Patwari, J. Croft, S. Jana, and S. K. Kasera, “High-rate uncorrelated bit extraction for shared secret key generation from channel measurements,” *IEEE Trans. Mobile Comput.*, vol. 9, no. 1, pp. 17–30, Jan. 2010.

[8] L. Cheng, B. E. Henty, F. Bai, and D. D. Stancil, “Doppler spread and coherence time of rural and highway vehicle-to-vehicle channels at 5.9 ghz,” in *Proc. IEEE GLOBECOM*, 2008, pp. 1–6.

[9] J. Camp and E. Knightly, “Modulation rate adaptation in urban and vehicular environments: Cross-layer implementation and experimental evaluation,” in *Proc. 14th ACM Int. Conf. Mobile Comput. Netw.*, 2008, pp. 315–326.

[10] X. Zhu, F. Xu, E. Novak, C. C. Tan, Q. Li, and G. Chen, “Extracting secret key from wireless link dynamics in vehicular environments,” in *Proc. IEEE INFOCOM*, 2013, pp. 2283–2291.

[11] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, E. Barker, S. Leigh, M. Levenson, M. Vangel, D. Banks, A. Heckert, J. Dray, and S. Vo, “A statistical test suite for random and pseudorandom number generators for cryptographic applications,” NIST Special Publication 800-22 Revision 1a, Apr. 2010, http://dx.doi.org/10.6028/NIST.SP.800-22r1a

[12] B. Azimi-Sadjadi, A. Kiayias, A. Mercado, and B. Yener, “Robust key generation from signal envelopes in wireless networks,” in *Proc. 14th ACM Conf. Comput. Commun. Security*, 2007, pp. 401–410.

[13] S. Gollakota and D. Katabi, “Physical layer wireless security made fast and channel independent,” in *Proc. IEEE INFOCOM*, 2011, pp. 1125–1133.

[14] H. Liu, Y. Wang, J. Yang, and Y. Chen, “Fast and practical secret key extraction by exploiting channel response,” in *Proc. IEEE INFOCOM*, 2013, pp. 3048–3056.

[15] P. Huang and X. Wang, “Fast secret key generation in static wireless networks: A virtual channel approach,” in *Proc. IEEE INFOCOM*, 2013, pp. 2292–2300.

[16] M. Wilhelm, I. Martinovic, and J. B. Schmitt, “Secret keys from entangled sensor motes: Implementation and analysis,” in *Proc. 3rd ACM Conf. Wireless Netw. Security*, 2010, pp. 139–144.

[17] F. Xu, Z. Qin, C. C. Tan, B. Wang, and Q. Li, “Imdguard: Securing implantable medical devices with the external wearable guardian,” in *Proc. IEEE INFOCOM*, 2011, pp. 1862–1870.

[18] I. Safaka, C. Fragouli, K. Argyraki, and S. Diggavi, “Exchanging pairwise secrets efficiently,” in *Proc. IEEE INFOCOM*, 2013, pp. 2265–2273.

[19] H. Liu, J. Yang, Y. Wang, and Y. Chen, “Collaborative secret key extraction leveraging received signal strength in mobile wireless networks,” in *Proc. IEEE INFOCOM*, 2012, pp. 927–935.

[20] H. Liu, J. Yang, Y. Wang, Y. Chen, and C. Koksal, “Group secret key generation via received signal strength: Protocols, achievable rates, and implementation,” *IEEE Trans. Mobile Comput.*, vol. 13, no. 12, pp. 2820–2835, Dec. 2014.

[21] Z. Yang, A. C. Champion, B. Gu, X. Bai, and D. Xuan, “Link-layer protection in 802.11i WLANS with dummy authentication,” in *Proc. 2nd ACM Conf. Wireless Netw. Security*, 2009, pp. 131–138.

[22] H. Wang, B. Sheng, and Q. Li, “Privacy-aware routing in sensor networks,” *Int. J. Comput. Telecommun. Netw.*, vol. 53, no. 9, pp. 1512–1529, 2009.

[23] F. Liu, X. Cheng, L. Ma, and K. Xing, “Sbk: A self-configuring framework for bootstrapping keys in sensor networks,” *IEEE Trans. Mobile Comput.*, vol. 7, no. 7, pp. 858–868, Aug. 2008.

[24] M. Edman, A. Kiayias, and B. Yener, “On passive inference attacks against physical-layer key extraction?” in *Proc. 4th Eur. Workshop Syst. Security*, 2011, Art. no. 8.

[25] S. Eberz, M. Strohmeier, M. Wilhelm, and I. Martinovic, “A practical man-in-the-middle attack on signal-based key generation protocols,” in *Proc. 17th Eur. Symp. Res. Comput. Security*, 2012, pp. 235–252.

[26] R. Shaltiel, “Recent developments in explicit constructions of extractors,” *Bulletin EATCS*, vol. 77, pp. 67–95, 2002.

[27] P. Elias, “The efficient construction of an unbiased random sequence,” *Ann. Math. Statist.*, vol. 43, no. 3, pp. 864–870, 1972.

[28] C. H. Bennett, G. Brassard, and J.-M. Robert, “Privacy amplification by public discussion,” *SIAM J. Comput.*, vol. 17, pp. 210–229, 1988.

[29] G. Brassard and L. Salvail, “Secret-key reconciliation by public discussion,” in *Proc. Int. Workshop Theory Appl. Cryptographic Techn. Adv. Cryptology*, 1993, pp. 410–423.

[30] J. Kim, "Iterated grid search algorithm on unimodal criteria," Ph.D. dissertation, Virginia Polytechnic Inst. State Univ., Blacksburg, VA, 1997.

[31] S. Jana and S. K. Kasera, "On fast and accurate detection of unauthorized wireless access points using clock skews," in *Proc. 14th ACM Int. Conf. Mobile Comput. Netw.*, 2008, pp. 104–115.

[32] S. N. Prmnath, S. Jana, J. Croft, P. L. Gowda, M. Clark, S. K. Kasera, N. Patwari, and S. V. Krishnamurthy, "Secret key extraction from wireless signal strength in real environments," *IEEE Trans. Mobile Comput.*, vol. 12, no. 5, pp. 917–930, May 2013.

[33] D. R. Hardoon, S. R. Szedmak, and J. R. Shawe-taylor, "Canonical correlation analysis: An overview with application to learning methods," *Neural Comput.*, vol. 16, pp. 2639–2664, 2004.

[34] A. V. Prokhorov, "Partial correlation coefficient," *Encylcopaedia Math.*, http://www.encyclopediaofmath.org/index.php?title=Partial_correlation_coefficient&oldid=14288, last accessed: 2016-5-10.

[35] G. Schwarz, "Estimating the dimension of a model," *Ann. Statist.*, vol. 6, no. 2, pp. 461–464, 1978.

[36] I. Csiszar and P. C. Shields, "The consistency of the BIC Markov order estimator," *Ann. Statist.*, vol. 28, no. 6, pp. 1601–1619, 2000.

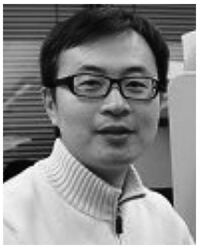
[37] M. Blum, "Independent unbiased coin flips from a correlated biased source: A finite state Markov chain," *J. Combinatorica*, vol. 6, no. 2, pp. 97–108, 1986.

[38] T. Calver, "An empirical analysis of the cascade secret key reconciliation protocol for quantum cryptography," Master Thesis, Air Force Inst. Technol., Wright-Patterson AFB, OH, 2011.

[39] M. Rice, *Digital Communications: A Discrete-Time Approach*. Englewood Cliffs, NJ, USA: Pearson/Prentice Hall, 2008.

[40] W. Jake, *Microwave Mobile Communications*. Hoboken, NJ, USA: Wiley, 1974.

Xiaojun Zhu received the BS and PhD degrees in computer science from Nanjing University in 2008 and 2014, respectively. He is an assistant professor in the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics. From August 2011 to August 2012, he was a visiting scholar at the College of William and Mary. His research interests include wireless networks, sensor networks, and vehicular networks. He is a member of the IEEE.



Fengyuan Xu received the PhD degree, with the Distinguished Dissertation Award, from the College of William and Mary. He is currently with the State Key Laboratory for Novel Software Technology, Nanjing University. His research interests include the broad areas of systems and security, with a focus on big data analytics for security, wireless device protection, energy efficient mobile systems, and graph and stream processing platforms. He is a member of the IEEE.

Edmund Novak received the BS degree from Monmouth College and the MS degree in computer science from the College of William and Mary, in 2010 and 2012, respectively. He is a fifth year student working toward the doctorate degree in computer science at the College of William and Mary. He is working with Dr. Quin Li, and his research interests include privacy and security on mobile devices and wearable computing.

Chiu C. Tan received the PhD degree from the College of William and Mary in 2010. He is an assistant professor in the Department of Computer and Information Sciences, Temple University. His research inetrests include the area of cyber security, mobile/cloud systems, smarthealth systems, and wireless network security (mainly 802.11, RFID, and sensor networks). He is a member of the IEEE.

Qun Li received the PhD degree in computer science from Dartmouth College. He is currently a professor in the Department of Computer Science, College of William and Mary. His research interests include wireless networks, sensor networks, pervasive computing, and security & privacy. He received the US NSF Career Award in 2008. He is a senior member of the IEEE.

Guihai Chen received the BS degree in computer software from Nanjing University in 1984, the ME degree in computer applications from the Southeast University in 1987, and the PhD degree in computer science from the University of Hong Kong in 1997. He is a distinguished professor at Shanghai Jiao Tong University. His research interests include parallel computing, wireless networks, data centers, peer-to-peer computing, high-performance computer architecture, and data engineering. He is a member of the IEEE.