KEEP: Fast Secret Key Extraction Protocol for D2D Communication

Wei Xi[1], Xiang-Yang Li[2,4], Chen Qian[3], Jinsong Han[1], Shaojie Tang[5], Jizhong Zhao[1], Kun Zhao[1]
[1] School of Electronic and Information Engineering, Xi'an Jiaotong University
[2] Department of Computer Science and Technology and TNLIST, Tsinghua University
[3] Department of Computer Science, University of Kentucky
[4] Department of Computer Science, Illinois Institute of Technology
[5] Department of Computer and Information Science, Temple University

***Abstract*—Device to device (D2D) communication is expected to become a promising technology of the next-generation wireless communication systems. Security issues have become technical barriers of D2D communication due to its "open-air" nature and lack of centralized control. Generating symmetric keys individually on different communication parties without key exchange or distribution is desirable but challenging. Recent work has proposed to extract keys from the measurement of physical layer random variations of a wireless channel, *e.g.*, the channel state information (CSI) from orthogonal frequency-division multiplexing (OFDM). Existing CSI-based key extraction methods usually use the measurement results of individual subcarriers. However, our real world experiment results show that CSI measurements from near-by subcarriers have strong correlations and a generated key may have a large proportion of repeated bit segments. Hence attackers may crack the key in a relatively short time and hence reduce the security level of the generated keys. In this work, we propose a fast secret key extraction protocol, called KEEP. KEEP uses a *validation-recombination* mechanism to obtain consistent secret keys from CSI measurements of all subcarriers. It achieves high security level of the keys and fast key-generation rate. We implement KEEP using off-the-shelf 802.11n devices and evaluate its performance via extensive experiments. Both theoretical analysis and experimental results demonstrate that KEEP is safer and more effective than the state-of-the-art approaches.**

I. Introduction

Device to device (D2D) communication provides technological support for local area services (local content sharing, mobile offline payment, *etc.*) that appear to be a promising component in next generation wireless communication systems. D2D communication is vulnerable to various attacks due to its "open air" nature and lack of centralized control. Cryptographic key establishment is a fundamental requirement of secure communication that supports confidentiality and authentication services, and is crucial for preserving user privacy [22] [24] [23] [13]. Achieving fast and reliable key agreement between wireless communication parties using a shared channel is challenging but desired due to its efficiency [12] [14] [2].

One recent trend in this regard is to allow two parties to build keys separately using inherent wireless channel properties [16]. Such secret key generation process consists of three technological components: random bit generation, information reconciliation, and privacy amplification [15] [17]. In random bit generation, a quantization method is used to convert measurement values of some "random" signal to information bits. A good quantizer (*e.g.*, [11]) can maximize the mutual information between two communicating entities (say, Alice and Bob) without information leakage. The information reconciliation process uses either error correcting codes [5] or interactive information reconciliation protocols (*e.g.*, Cascade [3]), to find and remove inconsistent bits in two bitstreams generated by the two end-parties. Privacy amplification [17] [19] [10] protect the confidentiality and privacy of key generation using cryptographical tools such as universal hash functions.

Existing key extraction protocols mainly use received signal strength (RSS) and other channel information of a single frequency to generate a bit stream as the key [21] [9] [7] [11]. Mathur *et al.* [16] propose a solution to extract a secret key from unauthenticated wireless channels using channel impulse response and amplitude measurements. Such channel randomness can also be exploited for device pairing [8] and authentication [20]. Extracting secret keys over MIMO was recently introduced in [18].

Using RSS to generate keys has the following drawbacks: (1) Bit generation rate is low, because each sample can only provide one RSS value. (2) It is vulnerable to predictable channel attacks, because RSS reading will increase and decrease if the channel is blocked periodically. (3) It cannot work well in static scenarios due to infrequent and small-scale variations in channel measurements.

To address these issues, recent studies propose to extract secret keys using channel state information (CSI) available from Orthogonal Frequency-Division Multiplexing (OFDM) [29]. Different from RSS, CSI is a fine-grained metric derived from the physical layer. It consists of 56 pairs of amplitude and phase belonging to 56 subcarriers in frequency domain, which can be utilized to achieve higher generation bit rate.

However, in our experimental study we find that adjacent or near-by subcarriers have similar physical characteristics, thus CSI measurements from them may have strong correlations.

978-1-4799-4852-9/14/$31.00 ©2014 IEEE

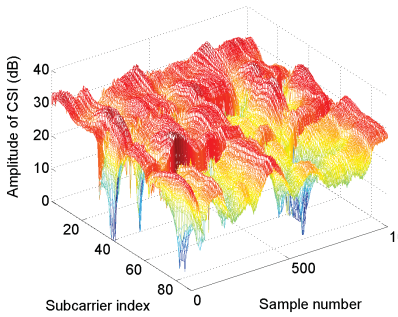
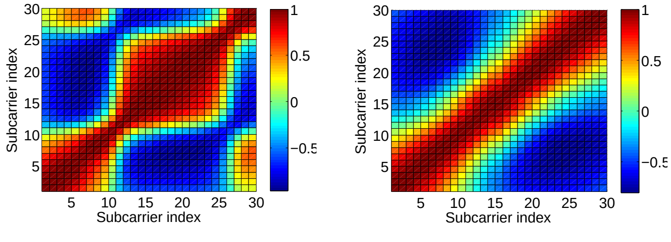


Fig. 1. Measured CSI values from all subcarriers using a NIC with 3 antennas by a user walking freely



(a) Static scenario

(b) Mobile scenario

Fig. 2. CSI measurements correlation among 30 subcarrieres

TABLE I
VARIABLE DEFINITIONS

Symbol	**Definition**
a, b	Alice, Bob
m	the number of subcarriers ($m = 30$ in our system)
B_{ai}	a bit stream of Alice generated from her i-th subcarrier
K_{ai}	a matched key of Alice generated from her i-th subcarrier
K'_{ai}	a mismatched bit stream of Alice from her i-th subcarrier

A key generated from near-by subcarriers may have a large proportion of repeated bit segments, which is a risk under key cracking attacks. Unfortunately, prior works pay little attentions to this problem. To address this issue, we propose a fast secret key extraction protocol, KEEP, which applies the *validation-recombination* mechanism to extract keys using the combined information of all subcarriers. It can efficiently prevent attackers from using correlations of subcarriers to crack secret keys, thus it achieves both high security level and fast key-generation rate. The contributions of this work are summarized as follows.

- We propose a mismatch federated filtration method in **KEEP** for communication entities to drop the inconsistent bits by exploiting the correlation of CSI measurements from multiple subcarriers. **KEEP** can detect the CSI measurements whose variation trend is different from others in the same packet, and thus reduces the bit mismatch rate and is conducive to information reconciliation.
- **KEEP** uses a universal hash function to validate the consistency of bit streams between the two communication entities. The function has two merits. One is that any two similar bit streams will have very different hashing results. Therefore a pair of wireless devices can validate the consistency of their keys with high accuracy by exchanging only small parts of their hash results. The other is that almost half bits of two streams are different even if they have similar hash results. It increases the difficultly for an attacker to figure out the original bit stream according to the part of the hash results transmitted in public channel.
- We apply a key recombination method to generate secret bit from a large number of subcarriers rather than a few near-by ones. Using adaptive quantization, the bit mismatch rate is low and the recombination method is more efficient than existing information reconciliation methods. Additionally, it eliminates the correlation of bit streams generated among multiple subcarriers and can resist the predictable channel attack as well.
- We formally analyze the efficiency and security of information reconciliation of **KEEP** and compare them with those of Parity-check. We also evaluate the performance of **KEEP** through extensive experiments using off-the-shelf 802.11n devices in real indoor and outdoor scenarios. Our evaluation shows that **KEEP** achieves high key generation rate and low bit mismatch rate and prevents information leakage in information reconciliation.

The rest of this paper is organized as follows. Section II presents the problem formulation and real world observations. The design of **KEEP** is elaborated in Section III, followed by theoretically analysis in Section IV. The performance of **KEEP** is evaluated in Section V. We conclude the paper in Section VI.

II. PROBLEM FORMULATION AND PRELIMINARY OBSERVATION

In this section, we formulate the problem and present the motivation to improve the current work based on empirical results.

A. Problem formulation

We introduce the model of key establishment for D2D communication. We assume that two users, *Alice* and *Bob*, want to build a shared secret key by measuring their communication channel. In this paper, we assume that the devices use OFDM based communication protocols, such as IEEE 802.11n. A third user, *Eve*, attempts to find this key by passive or active attacks.

CSI model: We use CSI measurements to generate a symmetric secret key between Alice and Bob over public channel. The channel measurement used for key generation must be consistent to the two users. In practice, wireless devices operate in half duplex, *i.e.*, each node can operate their transmitter or receiver but not simultaneously. Only one side can measure the received signal at a time. Fortunately, as long as the time between two directional channel measurements is much smaller than the channel coherence time, CSI measurements are still similar. In general, the received signal measured by Alice and Bob at time t can be expressed as

$$r(t) = s(t)h(t) + \omega(t) \tag{1}$$

where s is the known probe signal, h is the stochastic process (*i.e.* CSI), and ω is the noise process. Wireless multipath channels $h(t)$ are often modeled as having an echo-type impulse response. In the rest of the paper, we will only focus on the amplitude of the CSI value. For ease of presentation, we summarize the notations and symbols used in this paper in Table I.

Quantization: Suppose at a time sequence $[t_1, t_2, \cdots, t_n]$, the channel measurements made by Alice and Bob are two amplitude sequences of CSI, $S_a(t_1, ..., t_n)$ and $S_b(t_1, ..., t_n)$. Alice and Bob aim to convert these channel measurements into an identical bit stream, denoted as B_a and B_b as their cryptographic keys.

In this work we use the following quantizer:

1) Alice divides $S_a(t_1, ..., t_n)$ into small blocks, each of which contains κ sample values. S_b is processed similarly.
2) For each block, the user calculates two adaptive thresholds q_+ and q_- independently
$$\begin{cases} q_+ = \mu_{S(t_1,...,t_n)} + \alpha * \sigma_{S(t_1,...,t_n)} \\ q_- = \mu_{S(t_1,...,t_n)} - \alpha * \sigma_{S(t_1,...,t_n)}, \end{cases}$$
where μ and σ are the mean and standard deviation of $S(t_1, ..., t_n)$, and $\alpha \geq 0$ is a tuning constant.
3) Alice and Bob parse their CSI measurements, discard those CSI estimates that lie between q_+ and q_-, and maintain a list of indices to track the CSI estimates that are discarded. They exchange their lists of dropped CSI estimates and only keep the ones that they both decide not to drop.
4) Alice and Bob generate their bit streams by extracting a "1" or a "0" for each CSI estimate if the estimate lies above q_+ or below q_-.

B. Preliminary Observation

CSI can be collected using off-the-shelf 802.11n wireless NICs. We use two laptops (named Alice and Bob) to perform our experiments of D2D communication. We install Linux 802.11n CSI Tool and Intel 5300 wireless NICs to spread out the signal received by one antenna and extract 30 pairs of amplitude and phase CSI values from each antenna [6]. We use *ping* command to guarantee sufficiently small time intervals between two directional channel measurements. The initiator requests the receiver to immediately reply once receiving the ping message. The round-trip delay is around 5ms.

An OFDM channel is orthogonally divided into multiple subcarriers. Figure 1 shows the multipath fading on a mobile radio channel reflected in 90 subcarriers of three antennas. Figure 2 plots the correlation of CSI measurements among 30 subcarriers in static and mobile scenarios respectively. It shows that the CSI measurements have *strong correlation between two adjacent subcarriers* in both static and mobile scenarios. For adjacent subcarriers, the correlations of CSI values are usually more than 0.8. Although the subcarrier signals in an OFDM channel are orthogonal with different frequencies, the adjacent subcarriers have very similar frequencies which results in the similar channel responses at the receiver in the frequency domain.

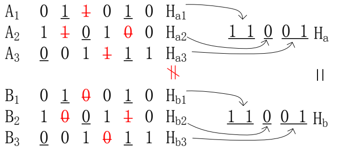


Fig. 3. "validation-recombination" mechanism

Based on the above observation, a key generated from adjacent subcarriers may have many identical segments, which could be a risk factor under key cracking. In this work, we propose to extract keys using the combined information of all subcarriers.

III. Key Extraction Protocol

The core design of our CSI-based secret key extraction consists of three components: *adaptive bit stream generation*, *leakage-resilient consistency validation*, and *key recombination*. Adaptive bit stream generation is used to convert measurement values to bits. Leakage-resilient consistency validation uses a universal hash function to check the consistency of generated bit streams on both sides. If inconsistent, then the bit streams will be recombined by randomly picking up some bits segments from different subcarriers until their hash results are the same.

A. Converting Measurements into Bits

We adopt an adaptive quantizer method to convert the CSI measurements to bits, as described in Section II-A. However, it is challenging to choose the proper threshold values q_+ and q_-. We conduct a set of experiments and show the CSI measurements in Figure 4, where Alice and Bob are both static. Two red dash lines denote the threshold q_+ and q_- respectively calculated from the whole CSI measurement values ($\alpha = 0.3$). We found that most values with sample index < 250 are smaller than q_-, while the values with sample index > 250 are mostly bigger than q_+. If fixed values of q_+ and q_- are used, it leads to long run of zeros in the front part and long run of ones in the rear part. It is undesired because this bit stream has low entropy and easy to break. Hence we allow each user to divide CSI samples into small blocks and then calculate the thresholds for each block separately to obtain bit streams with high randomness, especially in static scenarios.

In order to determine an appropriate block size, we conduct extensive experiments to investigate the relationship between block size and the number of inconsistent bits on two sides. In our experiments, we place Alice and Bob in different positions

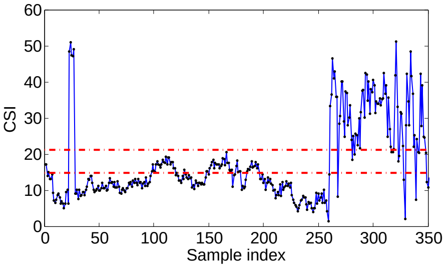


Fig. 4. CSI measurement in static scenario

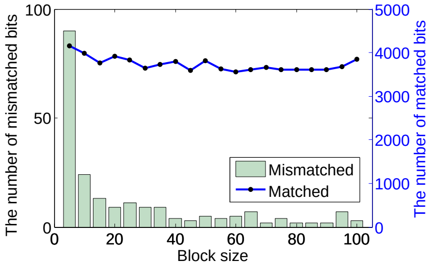


Fig. 5. Mismatch rate against block size κ

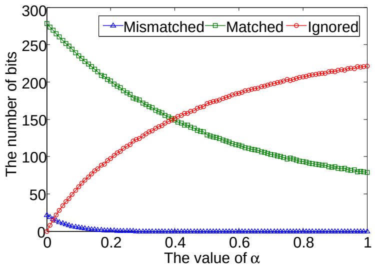


Fig. 6. The impact of α

and gather 217 samples of channel measurements, including 6510 CSI values from 30 subcarriers. The sampling interval is 200ms and $\alpha = 0.4$. Figure 5 shows the numbers of mismatched bits with different block sizes. We observe that if the block size is too small, the number of mismatched bits increases. It is because some outlier samples with extremely big or small values seriously influence the mean value of CSI in one block, which leads to inappropriate threshold q_+ and q_-. In our system, we set the block size $\kappa = 50$.

According to Figure 6, larger values of α result low mismatched bits ratios. However, a large α may slow down the bits generation rate. We find that using $\alpha \geq 0.3$ can reduce the bit errors to 0, and the matched bits are more than 50%. In our experiments, we set $\alpha = 0.45$ in mobile scenarios and $\alpha = 0.7$ in static scenarios.

B. Leakage-resilient Consistency Validation

Although federated filtration can reduce bit mismatch rate, we cannot use two bit streams as a shared key unless they are completely consistent. Existing reconciliation methods either use error correcting codes or some interactive information (*e.g.*, Hamming distance) reconciliation techniques to correct errors. However, these method may leak information to adversaries, especially considering that CSI values from subcarriers are correlated.

We propose a new ***validation-recombination*** mechanism to obtain fully consistent bit streams for key generation. The mechanism is illustrated in Figure 3. In the figure, Alice and Bob has 3 pairs of bit streams. There are some mismatched bits in each pair. They check the hash results (validation) and constantly regenerate a new bit stream (recombination) until the hash results are the same.

A family of hash functions is a collection of polynomial-time computable functions

$$\mathcal{H} = \left\{ \mathcal{H}_n : \{0,1\}^{l_{key}(n)} \times \{0,1\}^{l_{in}(n)} \to \{0,1\}^{l_{out}(n)} \right\}$$

where n is the security parameter, satisfying $l_{out}(n) < l_{in}(n)$. l_{in}, l_{out}, l_{key} are input length, output length and key size of the hash function, respectively. h_k denotes the function $\mathcal{H}_n(k)$ associated with the key $k \in \{0,1\}^{l_{key}(n)}$. A collision occurs when a pair (x,y) satisfying $x \neq y$ and $h_k(x) = h_k(y)$ for h_k.

In this work, we choose the hash function which satisfys the two conditions. 1) It should have a low probability of a hash collision. 2) Once the collision occurs, the two initial bit sequences are much different from each other. Therefore, we introduce two definitions to depict these two properties:

Definition 1: **Universal hash** Let $\mathcal{U}$ be a universe of keys, and let $\mathcal{H}$ be a finite collection of hash functions mapping $\mathcal{U}$ to $\{0, 1, \ldots, m-1\}$. $\mathcal{H}$ is universal if $\forall x, y \in \mathcal{U}$

$$\Pr\{h \in \mathcal{H} : h(x) = h(y) \wedge x \neq y\} = \frac{1}{m}$$

The universal hash function is anti-collision and useless for reversing the initial key from the hash results.

Proposition 1.1: Hash n keys into m slots in table T by h which is randomly selected from $\mathcal{H}$. Denote the number of collisions with a given key x as C_x, we have

$$E\left[C_x\right] < \frac{n}{m}$$

Proof: Let C_x be the random variable denoting total collisions of keys in T with x, and let

$$C_{xy} = \begin{cases} 1 & h(x) = h(y) \\ 0 & otherwise \end{cases}$$

We have $E\left[C_{xy}\right] = 1 \cdot \frac{1}{m} + 0 \cdot \frac{m-1}{m} = \frac{1}{m}$. Therefore,

$$\begin{aligned} E\left[C_x\right] &= E\left[\sum_{y \in T-x} C_{xy}\right] \\ &= \sum_{y \in T-x} E\left[C_{xy}\right] \\ &= \frac{n-1}{m} < \frac{n}{m} \end{aligned}$$

■

Proposition 1.2: Let K be the random variable uniformly distributed in $\{0,1\}^n$ and h be a universal hash function $\{0,1\}^n \to \{0,1\}^r$. Then the conditional entropy $H(K|h(K))$ has a lower bound $n - r$, *i.e.*, $H(K|h(K)) \geq n - r$.

Proof: By the definition of universal hash function and information entropy, we have $H(h(K)) \leq r$, and the equality hold up iff $h(K)$ follows uniform distribution [30]. Meanwhile, $H(K) = n$ because K is a uniformly distributed random variable. Then, we rewrite $H(K|h(K))$ by $H(K|h(K)) = H(K, h(K)) - H(h(K)) = H(K) - H(h(K)) \geq n - r$. ■

In other words, universal hash function has two important properties: 1) $\mathcal{H}$ has a good performance in collision avoidance for any input $\mathcal{U}$; 2) it is computationally infeasible to reversely calculate K from $h(K)$. And note that, it is very easy to design a universal hash function family using the method introduced in [25].

Definition 2: **Collision-resistant** For any n, we say that $\mathcal{H}$ is an (s, ε)-CRHF (collision-resistant hash function) if for every nonuniform A of size s,

$$\Pr\left\{k \leftarrow \{0,1\}^{l_{key}(n)} : A(k)\ outputs\ a\ collision\ for\ h_k\right\} < \varepsilon$$

If we choose proper parameters of (ϵ, s), two similar bit streams will have very different hashing results. Instead, if two hash results have few different bits, almost half of the bits of original bits streams are different from each other.

In general, the function satisfying two conditions above requires Alice and Bob to exchange only a few part of hash results to validate the consistency of two bit streams with high accuracy. This feature makes Eve hard to figure out the original bit stream from the part of the hash results transmitted in public channel.

C. Key Recombination

If the bit stream pairs extracted from all m subcarriers are inconsistent, they cannot be directly used as the secret key. In this case, we propose a fast key recombination method. The key idea is as follows. Alice (or Bob) randomly picks up l bits from the m bit streams and combines them into a new bit streams B_{ar} (or B_{br}). Alice and Bob randomly select the bits in same positions (the positions can be determined based on some pseudo-random number generator) from the corresponding bit stream. Since the mismatched bits rate is low, a pair of newly generated bit steams have a high probability to be matched. Alice and Bob sill check the bit streams via consistency validation described in last subsection. If B_{ar} is still different from B_{br}, Alice and Bob will recombine their keys until B_{ar} and B_{br} pass the consistency validation test.

IV. Analysis

In this section, we theoretically analyze and compare **KEEP** and parity-check based approaches in terms of security and efficiency of information reconciliation.

A. Performance of hash function

In **KEEP**, we apply universal hash functions to validate the consistency of generated bit streams. An efficient hash function can improve the validation accuracy and reduce the leakage of information. To this end, the function should satisfy the universal and collision-resistant conditions mentioned in Section III. Following these conditions, for the hamming distance d of two bit streams of length n, the corresponding consecutive m ($m < n$) bits in the two streams have a same hash result. Since the range of d is less than n, the variance of d is finite. We select the Gaussian distribution $X \sim \mathcal{N}(\mu, \sigma^2)$, which is defined as

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

to formulate the distribution. The Gaussian distribution has the largest entropy given its variance. Hence Gaussian curve is almost the optimal choice to resist attack when hash results are exposed. Taking account of $n \in N$, we exploit the discrete normal function instead.

$$p(d) = \begin{cases} \int\limits_{-\infty}^{d+0.5} f(x)dx & d = 0 \\ \int\limits_{d-0.5}^{d+0.5} f(x)dx & 0 < d < n \\ \int\limits_{d-0.5}^{\infty} f(x)dx & d = n \\ 0 & otherwise \end{cases}$$

Besides, we expect that the number of bit streams approaches its peak value(< 1) when $d = \mu$ in order to thwart the attackers. That means

$$P(\left[\frac{n}{2}\right]) = \int_{\mu-0.5}^{\mu+0.5} f(x)dx = \mathrm{erf}(\frac{1}{2\sqrt{2}\sigma}) < 1$$

Therefore, we can estimate the extremum of σ by error function.

The accuracy of validation using the hash function above is

$$1 - \frac{1}{\sqrt{2\pi}} \int_0^k exp\left\{-\frac{(t-k/2)^2}{2\sigma^2}\right\} dt$$

where k is the number of the different bits between Alice and Bob. Since k is much less than $\frac{n}{2}$, the validation accuracy is much higher.

As discussed in Section III, we need to find the universal hash functions with good collision resistant property. Fortunately, SHA-1 is enough to meet the requirements. In cryptography, the hash function SHA-1 takes a message of length less than 2^{64} bits and produces a 160-bit hash value, which is a U.S. Federal Information Processing Standard published by the United States NIST. Although a collision is found in 2006, it is not generally broken [31]. SHA-1 in particular has published techniques more efficient than brute force for finding collisions [32]. Therefore, we use SHA-1 to validate the consistency in our system. We conduce a set of experiments to evaluate the performance of SHA-1 function. To avoid high computation overhead, we segment the bit stream with a length of 16 bits and 32 bits, and generate the hash results. Each hash result has a 160-bit message digest. For each hash result H_i of 16-bit streams S_i ($i = 0 \sim 2^{16} - 1$), we only check their first m ($m = 4, 5, 6, 7, 8$) bits of hash results, array the source bit streams S_i with the same result with a same block sequence for Alice and Bob, respectively.

Figure 7 shows the relationship between the number of bit streams in one block and their Hamming distance. It follows the Gaussian distribution. Figure 8 is a magnified portion of Figure 7. From the figure we can see that, two bit streams with 1 to 5 mismatched bits have very low probability to have the same hash results. In summary, SHA-1 satisfies the feature we defined before.

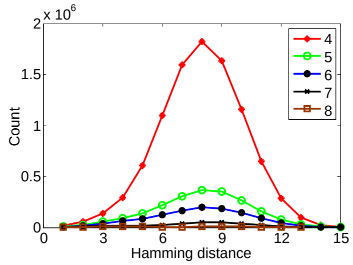


Fig. 7. Hamming distance distribution with the same hash result

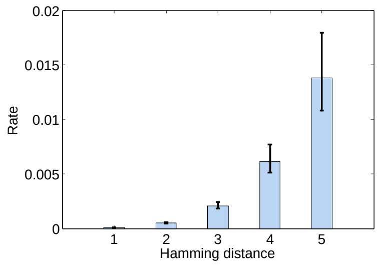


Fig. 8. The collision rate with different Hamming distance

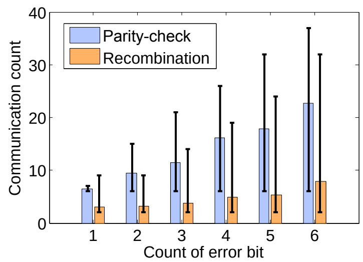


Fig. 9. Comparison of error correction efficiency between Parity-check and **KEEP**

B. Efficiency of information reconciliation

As we mentioned in the previous section, channel interferences lead inconsistency to the CSI measurement results among multiple subcarriers. To explore the efficiency of key generation, we analyze the expected number of rounds of information reconciliation as follows.

1) Parity-check: Parity-check can only discover odd bits error and correct one bit, and it is unable to find even bits error. Suppose N is the length of bit sequence and k is the number of errors, there are two cases to correct bit sequence errors using parity check.

(a) k is odd. The sequence is divided into two subsequence, and there must be odd errors in one subsequence. After $\log_2 N$ parity checks, we can locate one error and correct it. In other words, the expected count of correcting one error is $\log_2 N$, when k is odd.

(b) k is even. k_1 and k_2 are the number of errors in two subsequences. There are two situations, a) Both k_1 and k_2 are odd. In this case, we can correct two errors after $2 * \log_2(N/2) = \log_2 N$ parity check. b) Both k_1 and k_2 are even. In this situation, the errors cannot be discovered. Assume that the probability of the second situation is α. We have

$$\alpha = \frac{\sum_{i=1}^{k/2} C_{N-k}^{N/2-2i} C_k^{2i}}{C_N^{N/2}} \approx \frac{1}{2} \qquad (N > 5k) \tag{2}$$

The expectation count of locating and correcting one error is $2 \cdot (1-\alpha) \cdot \log_2 N + 0 \cdot \alpha = 2(1-\alpha)\log_2 N \approx \log_2 N$.

Therefore, to locate and correct one error of a N length bit sequence, it is expected to perform $\log_2 N$ parity check, no matter k is even or odd. It needs $k\log_2 N$ parity check to correct all k errors.

Suppose that the probability of success higher than a desired threshold $1-\beta$ needs t parity check. Then

$$1 - \beta < 1 - \alpha^t \tag{3}$$

We get $t > \log_\alpha \beta (\alpha \approx 1/2)$. If we set the significance level $\beta = 0.05$, t should not be less than 5.

In conclusion, with the significance level $\beta = 0.05$, if there are k errors, Parity check based methods need $k \cdot \log_2 N + 5$ times of comparisons.

*2) **KEEP**:* We choose l_i bits from subcarrier i to combine a key. In order to avoid the correlation between neighboring subcarriers, we select different sites between adjacent subcarriers. The match probability is $\Pr(l_i) = C_{L-d}^{l_i}/C_L^{l_i}$, where L expresses the sequence length and d denotes the count of mismatched bits. The match probability in r rounds is $1 - (1 - \prod_{i=1}^{m} \Pr_i(l_i))^r$, where m expresses the count of sequences. With a significance level of β, we have

$$1 - (1 - \prod_{i=1}^{m} \Pr_i(l_i))^r > 1 - \beta \tag{4}$$

$$r > \frac{\ln \beta}{\ln(1 - \prod_{i=1}^{m} \Pr_i(l_i))} \tag{5}$$

Since $\prod_{i=1}^{m} \Pr_i(l_i) << 1$, we have

$$r > \frac{\ln \beta}{\ln(1 - \prod_{i=1}^{m} \Pr_i(l_i))} \approx \frac{\ln(1/\beta)}{(\Pr(l))^m} \tag{6}$$

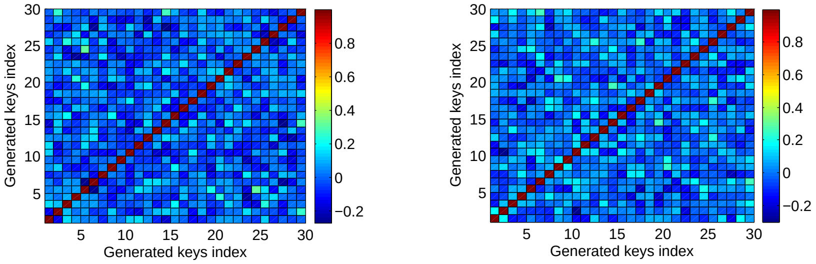
then

$$\frac{C_{L-d}^{l}}{C_L^{l}} > \sqrt[m]{\frac{\ln(1/\beta)}{r}} \tag{7}$$

We denote the channel bit error rate as e. Then we have $\frac{L!\sqrt[m]{\ln(1/\beta)}}{(L-L\cdot e)!} = C_1$, and

$$r > (C_1 \frac{(L-l-L\cdot e)!}{(L-l)!})^m \tag{8}$$

We conduct a simulation to evaluate the efficiency of our recombination method. We use 200 bits original streams to generate 150 bits keys. We increase the number of error bits from 1 to 6 of each subcarrier to test the recombination times. Figure 9 compares the error correction efficiency between Cascade and **KEEP**. The result shows that the mean amount of communications needed for the recombination of **KEEP** is lower than that of Cascade in all cases. Furthermore, this value of **KEEP** remains stable. For example, with 6 error bits, **KEEP** requires 8 times of communication in average, which is much less than the 24 times for Cascade.



(a) Correlation of **KEEP** keys in mobile scenario (b) Correlation of **KEEP** keys in static scenario

Fig. 10. Correlation of keys generated by existing methods and **KEEP**

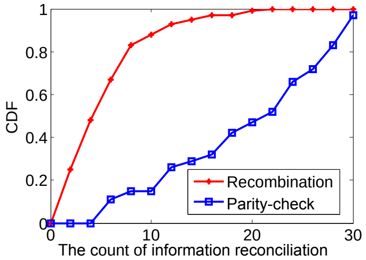


Fig. 11. The efficiency of information reconciliation

TABLE II
EXPERIMENTS SCENARIOS

Index	State	Environment
A	Static	Indoor
B	Static	Outdoor
C	Mobile	Indoor
D	Mobile	Outdoor

V. EVALUATION

A. *Methodology*

We conduct experiments with three laptops, named Alice, Bob and Eve, to evaluate the performance of **KEEP**. The laptops are all equipped with Intel 5300 wireless NICs, which are off-the-shelf products, running Linux with csitool kernel. Alice is configured as an AP. The wireless connection among three laptops operates in an 802.11n 2.4GHz channel. Their NIC clocks are synchronized. In this way, the CSI measurements can be aligned with time according to NIC clocks. In our experiments, Alice pings Bob every 100ms and receives Bob's ACK after 1-5ms. Eve is near to Bob, and turns into monitor mode to eavesdrop on the communication between Alice and Bob, *e.g.* recording the CSIs of their signals. The interval between two pings is 100ms because it is larger than coherence time. To reflect the performance of **KEEP** in real environments, we conduct experiments in four scenarios as shown in Table II.

To evaluate the performance of secret key extraction, we use the following metrics:

Bit Generation Rate: This metric partially reflects the speed of key generation. The bit generation rate is defined as the number of secret bits extracted from each packet after the secret bit quantization.

Bit Mismatch Rate: This metric shows the efficiency of generating keys. The bit mismatch rate is defined as the ratio of the number of mismatched bits to the total number of bits extracted via the secret bit quantization.

Randomness: It is to evaluate the distribution pattern of a binary sequence. Generally, the randomness of keys represents the quality of key generation methods. We measure the randomness of key generated by **KEEP** using the standard NIST test. In particular, we evaluate the correlation of keys generated from each subcarriers.

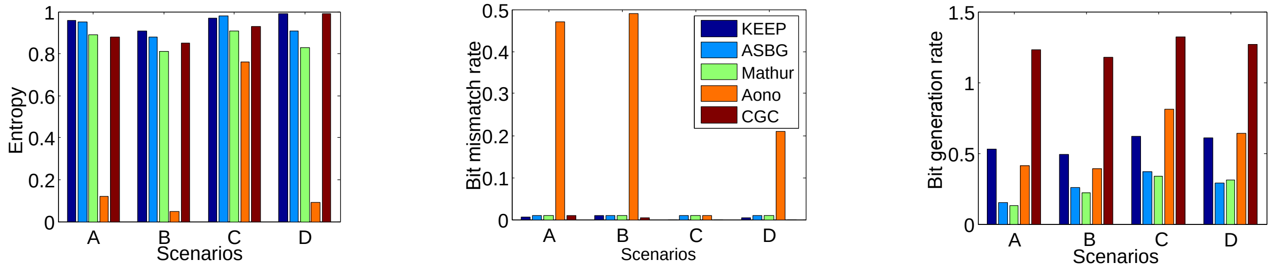
Bit Revealed Rate: We use the bit revealed rate to evaluate how much information will be leaked in the information reconciliation procedure. The bit revealed rate is defined as the ratio of the number of revealed bits in information reconciliation to the number of finally generated secret bits after information reconciliation.

We consider three attack models in our study. Due to limited space, we evaluate two attack, the fixed tracking attack and predictable channel attack in this paper.

B. *Correlation of generated key from different subcarriers*

Here we first demonstrate that previous key extraction approaches from CSI values of multiple subcarriers may result strong correlation among the bit streams, which can be utilized by the adversary to quickly crack the key. We evaluate the correlation of bit sequences generated key from different subcarriers of two typical quantization methods: quantization through the CSI variation with time in each subcarrier, and quantization based on the CSI diversity of multiple subcarriers [29].

Figure 2(a) and (b) plot the correlation of bits generated by quantization methods in mobile and static scenarios respectively. Since mismatched bits are discarded, we only use the parts of bit sequences identical on both side and calculate their correlations. The X and Y axis denote the subcarrier indices, and the color of grids denote the correlation of generated key of two subcarrier. The result shows that the keys have relatively strong correlation between close-by subcarriers, for both methods. The main reason of this phenomena is the principle of OFDM. OFDM encodes digital data into multiple carrier frequencies. The adjacent subcarriers have very similar frequencies which lead to the similar channel response at the receiver in frequency domain. Obviously high correlation among bit streams degrades the security of the generated keys. In order to ensure security, **KEEP** generates a secret key by bit stream recombination as described in Section III C, instead of directly using the bit streams generated from individual subcarriers as the keys. Figure 10(a) and (b) plot the correlation of keys generated by **KEEP**. The generated keys are not related to each other. That is because they consist of discrete and randomly fragments of each subcarrier, which



(a)

(b)

(c)

Fig. 12. Comparison of key extraction approaches: (a)Entropy comparison, (b)Bit mismatch rate comparison, (c)Bit generation rate comparison

can effectively combat the key-cracking attacks aim to the high correlation among bit streams.

C. Efficiency and Security of Information Reconciliation

As mentioned in Section IV C, consistency validation is performed to guarantee that the keys generated on both sides are identical. Such information reconciliation procedure includes multiple rounds of message exchanges until Alice and Bob find that they agree on a same key. Fewer rounds of message exchanges indicate a more efficient and secure key generation procedure, because in every round Eve can retrieval a little information in the eavesdropping. We evaluate the count of message exchange rounds in this subsection. We generate 30 different bit streams for each of Alice and Bob. Each stream contains 300 random bits. Initially, the streams are identical for Alice and Bob. In each simulation, we randomly select 1 to 3 bits to flip on each pair of streams. We then use our information reconciliation method for Alice and Bob to get identical keys. We measure the number of transmitted messages in the entire information reconciliation process of KEEP and Parity-check. Figure 11 reports the comparison between the two approaches. Clearly, **KEEP** outperforms Parity-check. For **KEEP**, more than 80% cases need less than 10 messages to achieve consistency, while Parity-check requires more than 28 messages between Alice and Bob in over 80 percent of the cases. In practice, most corresponding bits are consistent between two parties using **KEEP**, as long as a proper α value is selected. We may conclude that **KEEP**, using recombination of the mismatched bit streams generated from multiple subcarriers, is more efficient than the Parity-check method, which attempts to find out the mismatched bits and then correct them in the information reconciliation process.

As analyzed in Section IV, the secret bits are reduced one bit by running a parity check. Since parity checks are not independent, the secret bits are continuously reduced when we keep processing the checks. Differently, each hash-based validation used by **KEEP** is independent, the secret bits are only reduced by the last validation.

TABLE III
NIST STATISTICAL TEST SUITE RESULTS

Test	A	B	C	D
Frequency	0.731	0.575	0.854	0.758
Longest run of 1s	0.322	0.316	0.866	0.878
FFT	0.605	0.734	0.735	0.654
Approx. Entropy	0.588	0.829	0.609	0.671
Cum. sums (Fwd)	0.403	0.577	0.804	0.438
Cum. sums (Rev)	0.573	0.584	0.975	0.788
Runs	0.713	0.572	0.884	0.741

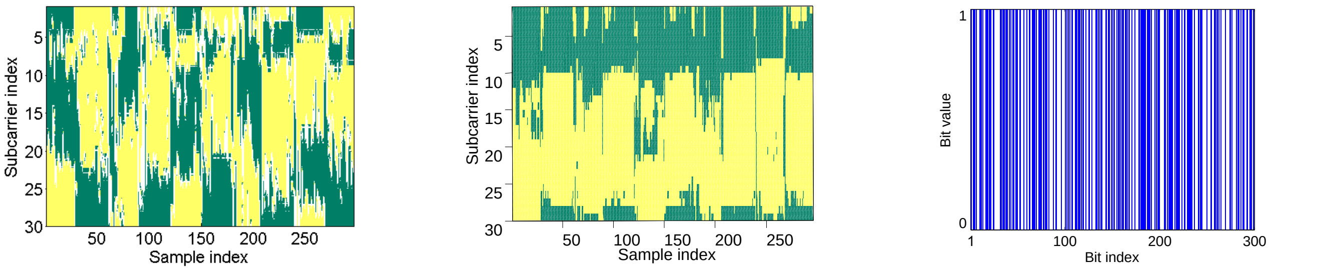
D. Randomness of Key

Ensuring the randomness of generated bits is crucial for key generation. We use the NIST statistical Test Suite to check the randomness of keys generated by **KEEP**. According to the specification in this suite, a p-value is the probability of obtaining a test statistic as large or larger than the one observed if the sequence is random. Hence, a smaller p-value indicates that the sequence is more unlikely to be random. Passing the NIST test requires a p-value larger than 0.01. We list the p-value of **KEEP** in 7 kinds of tests in Table III. From the result, we find that the bit streams generated by **KEEP** pass all the tests.

E. Comparison of Key Extraction Approaches

We compare **KEEP** with the existing typical key extraction approaches, *i.e.* Aono [28], Mathur *et al.* [16], ASBG [11], CGC [29]. For fairness, we align the baseline of comparison as follows. To implement Aono, the configurable parameter β is chosen such that at most 15% of the CSI measurements are dropped. In the scheme proposed by Mathur *et al.*, there are two parameters α and m. We set $\alpha = 0.35$ and $m = 2$ to ensure most fractions of measurements are used for bit extraction. For ASBG, CGC, and our **KEEP**, we choose $\alpha = 0.35$ and $block_{size} = 50$, where the mismatch rate is low.

The performance of the different secret key extraction approaches is shown in Figures 12. We calculate the entropy of keys generated by different approaches. The entropy can reflect the randomness of keys from the perspective of uncertainty. Aono has a very high secret bit generation rate. However it suffers from very low entropy. It is because Aono uses the median value of the measurements as a threshold



(a) Bits quantized from CSI variation with time

(b) Bits quantized from CSI diversity among multiple subcarriers

(c) Randomly variation of generated key of **KEEP**

Fig. 13. CSI measurements when an intermediate object moving between Alice and Bob.

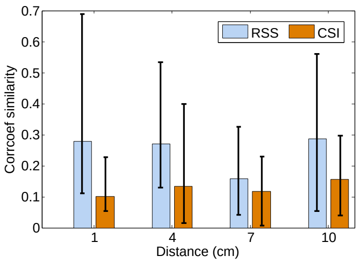


Fig. 14. Fixed tracking attack

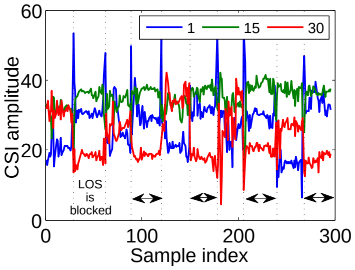


Fig. 15. Periodical variation of CSI amplitudes in three subcarriers

and drops any measurements that are close to the median value. As a consequence, the dropped bits in Aono is small and the secret bit rate is high. But the correlation among bits reduce the randomness of the key, as we discussed in Section V B. The work of Mathur *et al.* and ASBG have similar performance. Their entropy and bit mismatch rates are similar to those of **KEEP**, but the bit generation rates are much lower. CGC has the highest bit generation rate, because it utilizes the CSI measurements from all subcarriers to generate bit streams, which is highly efficient. However the keys are more vulnerable to cracking attacks as we demonstrated in Section V-B. **KEEP** generates bit streams with high entropy and bit generation rate than Mathur and ASBG. In particular, the bit mismatch rate of Mathur, ASBG, CGC and **KEEP** are almost zero in all scenarios. Based on above comprehensive results, **KEEP** performs a good trade-off between security and efficiency than other key extraction approaches.

F. Preventing Attacks

We evaluate the performance of key extraction of our protocol under predictable channel attack. Let a random variable W denote a random n-bit string held by Alice and Bob, while Eve learns a correlated random variable V, providing at most $t < n$ bits of information about W, *i.e.*, $H(W|V) \geq n - t$. The distribution P_{VW} are generally unknown to Alice and Bob.Eve's partial information on W and her complete information on function g give her arbitrarily little information about $K = g(W)$.If there are some correlation between Bob and Eve, Eve can generate the similar key as Bob's from its channel measurements.

1) Fixed Tracking Attack: In this experiment, Bob and Eve both remain stable with a certain distance and Alice move freely. Figure 14 compares the correlation of RSS and CSI measurements. For RSS measurements, the high correlation appears when Bob and Eve is about only 1cm away from each other. In this case, the maximum correlation is near 0.7 and the average is near 0.3. In contrast, the correlation of CSI measurements is independent from the distance between Bob and Eve. Even if the distance is within 1 cm, the maximum correlation is still less than 0.3 and average correlation is less than 0.1. The result shows that it is impossible for Eve to crack the key from the multipath fading channel used by Bob.

2) Predictable Channel Attack: The attacker Eve can perform planned movements to block the line-of-sight (LOS) between Alice and Bob such that the secret key extracted from the CSI measurements with desired changes becomes predictable when both Alice and Bob are stationary.

Figure 13 shows the variation of the CSI values under the predictable channel attack. The CSI values display periodical changes as shown in Figure 15. The CSI of 1st and 15th subcarriers increase when Eve blocks LOS and then decrease when Eve moves away. The opposite occurs in 30th subcarrier.

As mentioned before, there are two main previous quantization methods to convert CSI measurements into bits. The quantization results are shown in Figure 13 (a) and (b). The green and yellow parts represent "0" and "1" respectively. Obviously, the generated bits using both quantization methods have a certain distribution pattern. As such, the attacker can predict the changes of CSI measurements of Alice and Bob by observing intermediate objects blocking their LOS. That is, when the LOS between Alice and Bob is blocked and clear, the generated bit is inversion. Figure 13 (c) shows the generated key of **KEEP** under the predictable channel attack. The blue parts and white parts represent “1” and “0” respectively. Obviously, the variation of “0” and “1” are inconsistent with the blocking pattern. **KEEP** extracts keys by randomly picking up discrete fragments from all the subcarriers, which leads difficulty for the attacker in predicting identical secret bits to those of Alice or Bob by performing a predictable channel attack.

VI. Conclusion

In this paper, we propose a key extraction protocol, **KEEP**, that exploits CSI measurements to establish a shared secret key between two communication entities. Our protocol achieves high security level against various attacks including eavesdropping of passive attackers and the predictable channel attack from active attackers. **KEEP** adopts a validation-recombination algorithm to obtain a set of matched bit streams and eliminates the correlation of CSI measurements among subcarriers. Moreover, it significantly reduces the communication overhead and mitigates information leakage in the information reconciliation process. We implement **KEEP** using off-the-shelf 802.11n wireless NICs, and conduct extensive experiments to demonstrate the feasibility and efficiency of **KEEP**. In particular, **KEEP** provides a good trade-off among bit generation rate , mismatched bit rate, and entropy, in both static and mobile scenarios.

VII. Acknowledgements

This work is partially supported by the NSFC under Grant No. 61325013, 61033015 and 61373175, the Fundamental Research Funds for the Central Universities of China under Project No. 2012jdgz02 (Xian Jiaotong University), and the Research Fund for the Doctoral Program of Higher Education under project No. 20130201120016, NSF CNS-0832120, NSF CNS-1035894, NSF ECCS-1247944, NSF ECCS-1343306.

References

[1] 802.11n working group and others. IEEE 802.11n Specification 2009.

[2] Y. Amir, Y. Kim, C. Nita-Rotaru, J. Schultz, J. Stanton, and G. Tsudik. “Exploring robustness in group key agreement.” In *IEEE ICDCS*, pages 399–408, 2001.

[3] G. Brassard and L. Salvail. “Secret-key reconciliation by public discussion.” In *Advances in Cryptology-Eurocrypt*, pages 410–423. Springer, 1994.

[4] H. Chan, A. Perrig, and D. Song. “Random key predistribution schemes for sensor networks”. In *Symposium on Security and Privacy*, pages 197–213, 2003.

[5] Y. Dodis, L. Reyzin, and A. Smith. “Fuzzy extractors: How to generate strong keys from biometrics and other noisy data”. In *Advances in Cryptology-Eurocrypt*, pages 523–540. Springer, 2004.

[6] D. Halperin, W. Hu, A. Sheth, and D. Wetherall. “Tool release: gathering 802.11 n traces with channel state information.” *ACM SIGCOMM Computer Communication Review*, 41(1):53–53, 2011.

[7] A. Hassan, W. Stark, J. Hershey, and S. Chennakeshu. “Cryptographic key agreement for mobile radio.” *Digital Signal Processing*, 6(4):207–212, 1996.

[8] T. Heartbeats. “Proximate: Proximity-based secure pairing using ambient wireless signals.” *IEEE Wireless Communications*, page 8, 2011.

[9] J. Hershey, A. Hassan, and R. Yarlagadda. “Unconventional cryptographic keying variable management.” *IEEE Tran. on Communications*, 43(1):3–6, 1995.

[10] R. Impagliazzo, L. Levin, and M. Luby. “Pseudo-random generation from one-way functions.” In *ACM STOC*, pages 12–24, 1989.

[11] S. Jana, S. Premnath, M. Clark, S. Kasera, N. Patwari, and S. Krishnamurthy. “On the effectiveness of secret key extraction from wireless signal strength in real environments.” In *ACM MobiCom*, pages 321–332, 2009.

[12] P. Lee, J. Lui, and D. Yau. “Distributed collaborative key agreement and authentication protocols for dynamic peer groups.” *IEEE/ACM Tran. on Networking*, 14(2):263–276, 2006.

[13] M. Li, W. Lou, and K. Ren. “Data security and privacy in wireless body area networks.” *IEEE Wireless Communications*, 17(1):51–58, 2010.

[14] D. Liu, P. Ning, and R. Li. “Establishing pairwise keys in distributed sensor networks.” *ACM Trans. on Information and System Security*, 8(1):41–77, 2005.

[15] Y. Liu, S. Draper, and A. Sayeed. “Exploiting channel diversity in secret key generation from multipath fading randomness.” *IEEE Trans. on Information Forensics and Security*, PP(99):1, 2012.

[16] S. Mathur, W. Trappe, N. Mandayam, C. Ye, and A. Reznik. “Radio-telepathy: extracting a secret key from an unauthenticated wireless channel.” In *ACM MobiCom*, pages 128–139, 2008.

[17] U. Maurer and S. Wolf. “Secret-key agreement over unauthenticated public channels - part iii: Privacy amplification.” *IEEE Tran. on Information Theory*, 49(4):839–851, Apr. 2003.

[18] J. Wallace and R. Sharma. “Automatic secret keys from reciprocal mimo wireless channels: Measurement and analysis.” *IEEE Tran. on Information Forensics and Security*, 5(3):381–392, 2010.

[19] M. Wilhelm, I. Martinovic, and J. Schmitt. “On key agreement in wireless sensor networks based on radio transmission properties.” In *IEEE Workshop on Secure Network Protocols*, pages 37–42, 2009.

[20] L. Xiao, L. Greenstein, N. Mandayam, and W. Trappe. “Using the physical layer for wireless authentication in time-variant channels.” *IEEE Tran. on Wireless Communications*, 7(7):2571–2579, 2008.

[21] C. Ye, S. Mathur, A. Reznik, Y. Shah, W. Trappe, and N. Mandayam. “Information-theoretically secret key generation for fading wireless channels.” *IEEE Tran. on Information Forensics and Security*, 5(2):240 –254, june 2010.

[22] L. Zhou and Z. Haas. “Securing ad hoc networks.” *IEEE Network*, 13(6):24–30, 1999.

[23] S. Zhu, S. Setia, S. Jajodia, and P. Ning. “An interleaved hop-by-hop authentication scheme for filtering of injected false data in sensor networks.” In *IEEE Symposium on Security and Privacy*, pages 259–271, 2004.

[24] S. Zhu, S. Xu, S. Setia, and S. Jajodia. “Establishing pairwise keys for secure communication in ad hoc networks: A probabilistic approach.” In *IEEE ICNP*, pages 326–335, 2003.

[25] J.L Carter and M.N.Wegman. “Universal classes of hash functions,” *Journal of computer and system sciences*, vol.18, pp. 143-154, 1979.

[26] Shannon, C.E. “A mathematical theory of communication,” In *ACM SIGMOBILE Mobile Computing and Comm. Review*, vol.5, pages 3–55, 2001.

[27] CH Bennett, G. Brassard, C. Crrpeau, and UM Maurer. “Generalized privacy amplilication,” *IEEE Tran. on Information Theory*, vol.41, pp. 1915-1923, Nov. 1995.

[28] Aono, T. and Higuchi, K. and Ohira, T. and Komiyama, B. and Sasaoka, H. “Wireless secret key generation exploiting reactance-domain scalar response of multipath fading channels,” *IEEE Tran. on Antennas and Propagation*, vol.53, pages 3776–3784, 2005.

[29] H. Liu, Y. Wang, J. Yang, and Y. Chen. “Fast and Practical Secret Key Extraction by Exploiting Channel Response.” In *IEEE INFOCOM*, pages 3148–3156, 2013.

[30] R. Gray. “Entropy and information theory (Second Edition).” Springer, 1990.

[31] D. Christophe, and R. Christian. “Finding SHA-1 characteristics: General results and applications.” In *Advances in Cryptology-ASIACRYPT*, pages 1–20. Springer, 2006.

[32] X. Wang, Y. Yin, and H. Yu. “Finding collisions in the full SHA-1.” In *Advances in Cryptology-ASIACRYPT*, pages 17–36. Springer, 2005.