Manipulatable Wireless Key Establishment

Song Fang, Ian Markwood, Yao Liu
University of South Florida, Tampa, FL
{songf@mail, imarkwood@mail, yliu@cse}.usf.edu

***Abstract*—Due to the broadcast nature of the wireless medium, private communications are easily eavesdropped. This has spurred extensive research into secret key establishment using physical layer characteristics of wireless channels. In all these schemes, the shared secret keys directly originate from the physical features of the real wireless channel, which is highly dependent on the communication environment nearby. Also, previous schemes require performing information reconciliation, which increases both the costs of key establishment and the risk of key leakage over the not yet secured channel. In this paper, we exhibit a novel wireless key establishment method allowing the transmitter to specify arbitrary content as the secret key and cause the receiver to obtain the same key by using a channel manipulation technique. At the same time, eavesdroppers are prevented from deriving the secret key. We furthermore exploit the transmitter's ability to specify any content as the key by enabling the transmitter to apply error-correction code to the key. This means the receiver can automatically detect and correct any mismatched bits without sending key-related information back to the transmitter over the public channel. Experimental results demonstrate that the our key establishment method reaches a success rate as high as 91.0% for establishing a 168-bit key between the transmitter and the receiver, and meanwhile the chance that the eavesdropper can infer the key in meter-order range of the receiver is subdued into the range of 0∼0.10%.**

I. Introduction

Wireless key establishment has been widely studied in the past years (e.g., [1]–[12]) for its easy implementation, low computational requirement, and small energy consumption. The common intuition is to establish a shared key utilizing the fact that the transmitter and receiver of one wireless link can observe the same channel simultaneously, a property known as *wireless channel reciprocity*. Next, the *spatial uncorrelation property* of the wireless channel provides the security basis for these wireless key establishments, i.e., a receiver will observe differing channels between transmitters in different locations. Hence, an eavesdropper able to receive the signal sent by transmitters Alice or Bob will be unable to decrypt it, as the extracted channel characteristic will differ from that visible to Alice and Bob, so long as the eavesdropper is not co-located with either character.

Existing wireless key establishment techniques normally entail three steps to share a secret key between Alice and Bob, namely *quantization*, *reconciliation*, and *privacy amplification*. Quantization involves both parties sampling the channel characteristic and then quantizing the sampled data into initial binary bit sequences. Unfortunately, channel noise may cause the quantization step to render some moderately different bit sequences for Alice compared to Bob. Reconciliation schemes aim to correct these mismatched bits through information exchanges. Finally, each communicator performs privacy amplification to confuse a malicious listener from deducing the secret bit sequence.

This workflow naturally imparts two major drawbacks. First, the shared key originates from the actual channel, and the communicators cannot control such a key. The established key depends on the wireless channel dynamics, and a static environment results in an established key of low entropy [3]. Second, Alice and Bob must exchange multiple messages over the public channel to agree on an identical key during reconciliation. These messages contain sensitive key-related information and not only create an opportunity for an attacker to capture the exchanged information and infer the key [2], but also highly decrease the efficiency of key establishment. In this paper, we evolve fundamental aspects of existing key establishment techniques to eliminate these aforementioned deficiencies, by removing the mutual message exchange process heretofore universally required. Specifically, we enable Alice to deliver a key to Bob by sending a one-time message, once Alice receives a key establishment inquiry from Bob. Bob is then passive and no longer needs to make any unprotected communications.

Intuitively, this sort of key establishment scheme can be achieved if the transmitter can manipulate the channel characteristics observed by the receiver. This transmitter will select any random content as the key and generate channel characteristics equal to the key for the receiver to observe. Because the transmitter can specify any content as the key, the transmitter can further encode the key with error-correction code, and accordingly the receiver will be able to automatically correct any mismatched bits without sending key-related information to the transmitter over the public channel.

The spatial uncorrelation property is caused by the multipath propagation phenomenon, where a signal travels in the air over multiple paths due to signal reflections, diffractions, and scatterings. Geographically separated transmitter and receiver pairs encounter different multipaths and necessarily different channel characteristics. Hence, if we can create an "artificial multipath" effect, then we can manipulate the channel characteristics observed by the receiver, and specify "artificial" channel characteristics equal to the encoded key at the receiver.

Figure 1(a) shows a simple multipath propagation example involving three paths. The transmitter sends a wireless signal. The receiver observes the superposition of three signal copies, each of which is distorted by the corresponding path. We use h_i to denote the distortion introduced by Path i. In Figure 1(a),

978-1-5386-0683-4/17/$31.00

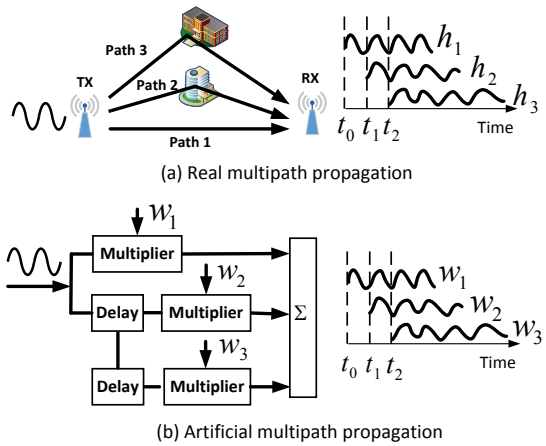

Fig. 1. Real v.s. artificial multipath effect.

the vector $[h_1, h_2, h_3]$ represents the channel characteristics of the 3-path channel and this vector is referred as the *channel impulse response* [13]. The shared key is normally quantized from the channel impulse response [2], [8], [10]. The signal copy distorted by Path i can be usually modeled by $h_i x$, where x is the original, undistorted signal [13]. It is easily understood then that multipath propagation in the real world can be simulated using simple delay and multiplication operations. Figure 1(b) illustrates how this may be accomplished. Here, the transmitter sends the original signal as well as time-delayed copies of this signal to mimic the different arrival times of each multipath component. The original signal and all copies are multiplied against coefficients w_i to mimic the signal distortion caused by each path i. Consequently, the receiver observes an aggregation of one signal plus time-delayed copies, each undergoing a certain path distortion, and thus obtains a channel impulse response vector of $[w_1, w_2, w_3]$ corresponding to the expected multipath effect but created entirely by the transmitter. The delay (e.g., $t_1 - t_0$) is the arrival time difference of two consecutively received signals. The challenges in order to build the proposed scheme are summarized below.

First, to specify the chosen channel impulse response at the receiver, the transmitter must cancel the real multipath effect without compromising the spatial uncorrelation property, which serves as the security foundation for all wireless key establishment techniques. To address this, we create a customized channel manipulation technique.

Second, upon receiving the signal, the receiver estimates and quantizes the channel impulse response to obtain the key. Hence, the transmitter must represent the key in the form of channel impulse responses, and find an appropriate key mapping method to enable the receiver to achieve a low error rate after the quantization. Accordingly, we design a key mapping technique to translate a key into manipulated channel impulse responses of a multipath channel.

Third, an important question is whether an eavesdropper can decode the key with the help of the same error correction code. When an eavesdropper's channel is uncorrelated with the receiver's channel, the key observed by the eavesdropper exhibits more deviation from the actual key than the receiver, allowing the ability of the communicators to choose a code type, which can resolve a number of bit errors larger than the receiver typically encounters but smaller than the number by the eavesdropper. For an eavesdropper located 4 meters away from the receiver, a 168-bit key can be established between two parties with a success rate as high as 91.0% at the receiver, while the probability that the eavesdropper can break the key is subdued into the range of 0∼0.1%.

The rest of the paper is organized as follows. Sections II describes preliminaries. We introduce assumptions and the attack model in Section III. The detailed proposed key establishment is then presented in Section IV, with experimental evaluation results in Section V. In Section VI, we summarize related work, and Section VII concludes the paper.

II. Preliminaries

Channel Estimation. Channel impulse response quantifies the effect of the multipath environment in wireless communications. Each path imposes a time delay, magnitude attenuation, and phase shift on the signal traveling along it. Channel is usually estimated based on training bit sequences and received signal samples. Physical layer channel estimation can be processed in either frequency or time domains which are inter-convertible due to their linear relationship [13].

The received signal $y(t)$ can be denoted as the convolution of the transmitted signal $x(t)$ and the channel impulse response $h(t)$ (we omit the noise term for the sake of simplicity): $y(t) = x(t) * h(t)$. In the frequency domain, we have $Y(f,t) = X(f,t)H(f,t)$, where $Y(f,t)$, $X(f,t)$ and $H(f,t)$ are the Fourier transforms of $y(t)$, $x(t)$ and $h(t)$, respectively. Thus, with knowledge of the transmitted and received signals, we easily obtain $H(f,t)$ and perform the inverse Fourier transform operation on it to find the corresponding channel impulse response $h(t)$, denoted as $h(t) = \mathcal{F}^{-1}\{\frac{Y(f,t)}{X(f,t)}\}$, where $\mathcal{F}^{-1}\{\cdot\}$ indicates the inverse Fourier transform. We sample the received signal with a symbol period of T_s and obtain the following sampled impulse response vector with a length of L: $h = [h_1, \cdots, h_L]$, where $h_i = h((i-1)T_s)$, $i \in \{1, \cdots, L\}$.

Error Correction Code (ECC). ECC is developed to correct errors in data transmission and commonly takes the form of block or convolutional ECC [13]. In block ECC, parity bits follow the information bits, while in convolutional codes they are interspersed together. Reed-Solomon (RS) ECC [14] is a typical block ECC with very strong error-correction capacity, especially against the burst errors inherent to wireless communications. Without loss of generality, we choose RS ECC to encode and reconstruct our keys.

III. Assumptions and Attack Model

In a general scenario, Alice wants to establish a secret key with Bob. We assume that Alice and Bob reside within each other's communication range and have the ability to do channel estimation. Also, we assume the training sequence for channel estimation is public, which conforms with the design of many commercial wireless communication systems [15]. Besides, before launching channel manipulation based key establishment, we assume Alice knows the actual channel impulse response between herself and Bob. This can be achieved

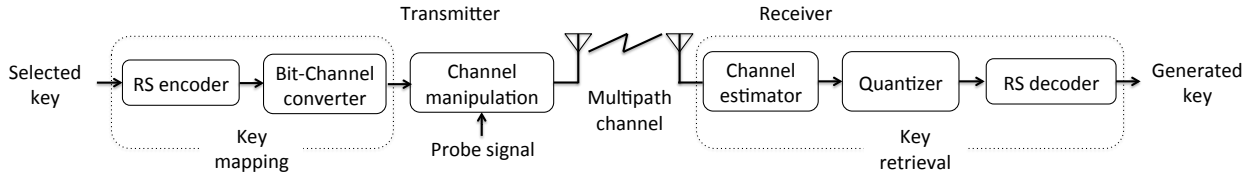

Fig. 2. Flow chart of manipulatable wireless key establishment.

by estimating the channel impulse response from the wireless signals (e.g., key establishment inquiries) emitted by Bob. Finally, the RS code enforced by Alice is assumed publicly available so that Bob can do the corresponding decryption.

In our attack model, we consider that an attacker Eve aims to derive the secret key established between Alice and Bob, and presume that Eve has the ability to 1) do channel estimation; 2) know the secret key quantization algorithm and the RS code that the transmitter utilizes.

IV. Key Establishment Scheme

A. Scheme Outline

The transmitter first chooses a key, encodes it with RS ECC. The selection of RS code is based on the length of the chosen key and other environmental factors which are studied in Section V-D. After that, the transmitter maps the encoded key into an artificial channel impulse response, denoted as $h'(t)$. To launch the channel manipulation, the transmitter needs then to calculate the aggregate signal $x'(t)$ to send. Correspondingly, the signal $y'(t)$ received from the transmitter can be represented by $y'(t) = x'(t) * h(t)$, where $h(t)$ is the actual channel impulse response between the transmitter and the receiver. The receiver uses $y'(t)$ and the public transmitted signal $x(t)$ to do channel estimation. With $\hat{h}(t)$ denoting the estimated channel impulse response at the receiver, we have $x(t) * \hat{h}(t) = x'(t) * h(t)$. Hence, the transmitter should construct the aggregated signal $x'(t)$ to make $\hat{h}(t) = h'(t)$. We give the detailed calculation in Section IV-C.

With $\hat{h}(t)$, the receiver first obtains a bit sequence via quantization, and then feeds it into a RS decoder. As long as the number of symbol errors in the quantized bit sequence does not exceed the error correction capability of the chosen RS code, the receiver can successfully recover the secret key specified by the transmitter. Figure 2 shows the flow chart of the proposed key establishment.

B. Key Mapping

Key mapping is the process of encrypting the selected binary key bits with RS code, and converting the encrypted key into an artificial channel impulse response. A channel impulse response is actually composed of a sequence of complex numbers. To simplify the conversion process, the channel impulse response used here refers to its magnitude, which is a vector of decimal numbers. Thus, key mapping can be regarded as a binary-decimal conversion.

The transmitter can arbitrarily specify a sequence of bits as the secret key, and then input it into an RS encoder. An RS(n, k) code has n symbols of s bits each, the first k of which are symbols comprised of selected key bits and any required padding and the rest calculated based on the RS algorithm and the k-symbol input. Given a symbol size s, the maximum length of the encoded message is $m = 2^s - 1$, so $n \leq m$ should hold. Finally, the RS decoder can correct any $(n-k)/2$ symbol errors in the encoded message. As an example, RS(31, 15) code with 5 bits for each symbol can accommodate a chosen key size of up to $15 \times 5 = 75$ bits, where if less than 75, remaining bits are padded with zeros. Also, errors in up to $(31 - 15)/2 = 8$ symbols anywhere in the encoded message can be corrected by the decoder.

The next step is to convert the encoded message with $n * s$ bits into channel impulse responses, each a length-L vector of decimal numbers, where L denotes the maximum number of manipulated resolvable multipath components that can be observed by the receiver. Generally, the transmitter can use two different conversion strategies, an absolute value based and a relative value based. In the former, a bit sequence is regarded as a binary number and directly translated into a decimal number denoting the value of a path response. In the latter case, a bit is translated into a quantitative relationship of two path responses, and this relationship may be the comparison result of the value size or the ratio. In this paper, we focus on the relative value based method, mapping the bits into the relation (difference) between path response values.

Let $h_m = [h_{m1}, h_{m2}, \cdots, h_{mL}]$ denote the generated channel impulse response. The differences between the first path response value and the others are $\Delta_1 = h_{m2} - h_{m1}$, $\Delta_2 = h_{m3} - h_{m1}$, $\cdots$, $\Delta_{L-1} = h_{mL} - h_{m1}$ respectively. With these $(L-1)$ path response value differences, we can compute another $\lfloor (L-1)/2 \rfloor$ differences, defined by $\Delta_{(L-1)+i} = |\Delta_{2i-1}| - |\Delta_{2i}|, i \in \{1, 2, \cdots, \lfloor (L-1)/2 \rfloor\}$. In fact, further levels could be utilized, but we use two for the scope of this paper. Hence, a bit sequence with length $L_b = L - 1 + \lfloor (L-1)/2 \rfloor$ can generate a channel impulse response with length L using each of these differences to define one bit. In order to reduce the error, the transmitter uses a quantum q as a positive or negative *path differential* to distinguish these differences for constructing an artificial multipath response. Based on the selected key and this q, the manipulated channel is assembled such that

$$\begin{cases} \Delta_i = q_+, & if\ key\ bit\ is\ 1 \\ \Delta_i = q_-, & if\ key\ bit\ is\ 0, \end{cases}$$

where $i \in \{1, 2, \cdots, L_b\}$. In Section V-C, we show that when the path differential q is well chosen, the bits extracted by the receiver are identical to the bits specified by the transmitter with high probability. In this manner, the encoded key of $n * s$ bits can be mapped into $\lceil (n * s)/L_b \rceil$ channel impulse responses, which are then launched in the channel manipulation step which follows.

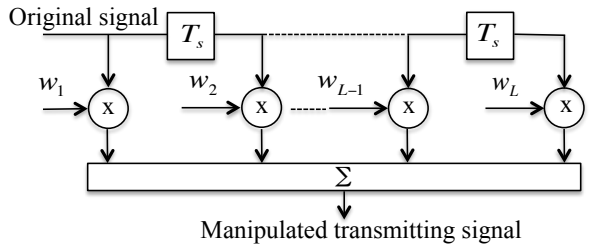

Fig. 3. Channel manipulation filter.

C. Channel Manipulation

The goal of channel manipulation is to make the channel impulse response estimated at the receiver equal to the artificial channel impulse response generated through key mapping, i.e., $\hat{h} = h_m$. As mentioned earlier, the transmitter intends to emulate the real multipath effect by sending aggregated weighted, delayed copies of the original transmitting signal.

Again, channel manipulation can be regarded as a delay-weight-sum module, implemented by a finite impulse response (FIR) filter as shown in Figure 3. The delay is set to one symbol duration, the transmission time of one symbol as denoted by T_s. The impulse response of the channel manipulation process can be represented by $w(t) = \sum_{i=1}^{L} w_i \delta(t - t_0 - (i-1)T_s)$, where t_0 is the arrival time of the first arrived signal copy. Thus, the manipulated transmission signal is the convolution of the transmitted signal $x(t)$ with the impulse response $w(t)$. This manipulated transmission signal $x(t) * w(t)$ is sent to the receiver through the real multipath channel. Hence, the corresponding signal obtained by the receiver is $y_m(t) = (x(t) * w(t)) * h(t)$. The receiver then utilizes the received signal and the public probe signal $x(t)$ to estimate the channel impulse response, described as $y_m(t) = x(t) * \hat{h}(t)$. Based on the associativity of convolution, we obtain $\hat{h}(t) = w(t) * h(t)$. Channel manipulation aims to make $\hat{h}(t) = h_m(t)$, where $h_m(t)$ is the value of h_m at a given time. Therefore, we have $h_m(t) = w(t) * h(t)$, and in the frequency we get $H_m(f,t) = W(f,t)H(f,t)$, where $H_m(f,t)$ and $W(f,t)$ are the Fourier transforms of $h_m(t)$ and $w(t)$. Therefore, we can solve the impulse response $w(t)$ of channel manipulation as

$$w(t) \quad = \quad \mathcal{F}^{-1}\{\frac{H_m(f,t)}{H(f,t)}\} = \mathcal{F}^{-1}\{\frac{Y_m(f,t)}{Y(f,t)}\}.$$

Thus, the transmitter is able to set the parameter $w = [w_1, \cdots, w_L]$, enabling the receiver to obtain the generated channel impulse response h_m.

D. Key Retrieval

With the estimated channel impulse response, the receiver runs quantization to generate bits. This key retrieval process is an inverse process of key mapping, i.e., a decimal-binary conversion. The quantizer we use is defined as

$$Q(\Delta_i) = \begin{cases} 1, & if\ \Delta_i > 0 \\ 0, & otherwise. \end{cases}$$

Note that since we remove the exchange of reconciliation information, missing bits are not handled in quantization. The quantizer defined above always returns a value, so the receiver will obtain a bit sequence of length L_b after applying quantization to an estimated channel impulse response.

- With obtained channel estimates $\hat{h} = [\hat{h}_1, \hat{h}_2, \cdots, \hat{h}_L]$, the receiver first calculates the $L-1$ differences between the first estimated path response and each of the others with the equation $\Delta_i = \hat{h}_{i+1} - \hat{h}_1$, where $i \in \{1, 2, \cdots, L-1\}$. For each calculated difference, the receiver quantizes it using $Q(\cdot)$.
- The receiver next computes the $\lfloor (L-1)/2 \rfloor$ differences between every pair of differences calculated in the previous step, using $\Delta_{L-1+i} = |\hat{\Delta}_{2i-1}| - |\hat{\Delta}_{2i}|$, where $i \in \{1, 2, \cdots, \lfloor (L-1)/2 \rfloor\}$. Again, the receiver quantizes each newly obtained difference with quantizer $Q(\cdot)$. Between these two steps, the total number of differences calculated is $L_b = L - 1 + \lfloor (L-1)/2 \rfloor$.
- Once the length of the obtained bits reaches the required length $n * s$ of the chosen RS(n,k) decoder, the receiver decodes them, and the first k symbols in the decoded message are then extracted as the secret key of $k * s$ bits.

To more clearly demonstrate the key retrieval process, we pick $L = 3$ as an example. At the receiver side, the estimated channel impulse response is denoted as $\hat{h} = [\hat{h}_1, \hat{h}_2, \hat{h}_3]$, the corresponding differences are calculated by $\Delta_1 = \hat{h}_2 - \hat{h}_1$, $\Delta_2 = \hat{h}_3 - \hat{h}_1$ and $\Delta_3 = |\Delta_1| - |\Delta_2|$. The receiver then obtains 3 bits by quantizing. This process is repeated, reassembling the encoded key 3 bits at a time, until all (possibly including padding) have arrived. The bits are finally decoded according to the selected RS ECC scheme into the secret key.

E. Security Analysis

1) Eavesdropping attacks: The correlation coefficient ρ between the two channels observed at the receiver and the eavesdropper respectively can be present with $J_0(2\pi \frac{D}{\lambda})$, where $J_0(\cdot)$ is the first kind Bessel function of order zero, and D is the distance between the receiver and the eavesdropper [16], [17]. When $D/\lambda \geq 1/2$, ρ approaches 0. That means, when the eavesdropper and the receiver are spaced at least a half wavelength apart, it yields zero correlation. However, in the real world, there is poor scattering and/or a strong line-of-sight component, the spatial separation between the receiver and the eavesdropper should be longer (e.g., several wavelengths) in order to obtain uncorrelated channels [18]. Generally, the eavesdropper may employ two different schemes to obtain the manipulated channel (and thus extract the secret key):

Direct Observation: It is unlikely that the eavesdropper could occupy the same physical location with the transmitter or receiver, as the exposure risk would be dramatically increased. Due to the spatial uncorrelation property, a small location difference (e.g., several wavelengths) would cause an observed channel change, and following discrepancies in the extracted keys.

Indirect Observation: A further concern is whether it is possible for the eavesdropper to calculate the manipulated channel with her own observed channel even she is not at the same location with the receiver. As mentioned in Section IV-C, the manipulated channel $h_m(t) = w(t) * h(t)$. So to learn

$h_m(t)$, an eavesdropper should not only know the impulse response $w(t)$ of the channel manipulation process, but also the real channel impulse response $h(t)$ between the transmitter and the receiver.

Let $y_e(t)$ denote the signal received by the eavesdropper when the transmitter launches channel manipulation, and let $h_e(t)$ denote the real channel impulse response between the transmitter and eavesdropper. Thus we have $y_e(t) = x(t) * w(t) * h_e(t)$. Therefore, to learn $w(t)$, the eavesdropper must learn $h_e(t)$. However, the transmitter can always hide its real channel or stay silent before launching key establishment, thus the eavesdropper would fail to obtain $w(t)$. Besides, we allow the transmitter to randomize the channel manipulation process and thus the value of $w(t)$ can be updated at any time, so that the eavesdropper will require far more effort. Even $w(t)$ is disclosed, $h(t)$ is also unknown to the eavesdropper due to the same reason as in the first scheme (i.e., the eavesdropper is unable to put a helper node co-located with the transmitter to measure the real channel). Therefore, this attack requirement is more stringent than the previous.

2) RS code impact: Due to the noise and hardware differences, the quantized bit sequence at the receiver may have bit discrepancies with the key K_a selected by the transmitter. With RS code, the receiver can obtain the secret key K_b and $K_b = K_a$. One concern is whether the enforced RS code can help the eavesdropper to correct those mismatched bits as well, leading that her extracted key K_e is same with K_a. We define a term *channel proximity*, denoted with d_{ij}, to quantify the difference between two channels i and j, which equals the Euclidean distance between two obtained channel impulse responses, i.e., $d_{ij} = ||h_i - h_j||$. Channel proximity between respectively observed channels at an eavesdropper and a receiver highly depends on the distance between them.

Suppose C denotes the count of mismatched symbols in the quantized symbol sequence (e.g., $\{s_1, \cdots, s_n\}$) for a corresponding n-symbol codeword (e.g., $\{s^1_{enc}, \cdots, s^n_{enc}\}$) selected by the transmitter. We construct a function $M(\cdot)$ to model the relationship between the channel proximity d_{ij} with the count C, i.e., $C = M(d_{ij}) = \sum_{j\in\{1,\cdots,n\}}(s^j \oplus s^j_{enc})$ (if s^j and s^j_{enc} have mismatched bits, $s^j \oplus s^j_{enc}$ equals 1, otherwise 0). Theoretically, when $d_{ij} = 0$, i.e., the two observed channels are totally the same, the generated bit sequences from them should be the same, i.e., $C = 0$. While if $d_{ij} = \infty$, i.e., the two target channels are totally different, the worst case is that all symbols in the generated two bit sequences are different, i.e., $C = n$. We utilize d_{tr} and d_{te} to denote the channel proximities between the estimated channels at the transmitter and the receiver, and at the transmitter and the eavesdropper, respectively. In the experiment (i.e., Section V-C), we show that $M(\cdot)$ is a monotonically increasing function in practice. Lemma 1 presents that with an appropriately selected RS code, the transmitter and the receiver are able to establish a secret key in the presence of an eavesdropper.

Lemma 1. *Suppose that $M(\cdot)$ is monotonically increasing, when the selected $RS(n,\ k)$ satisfies $d_{tr} \le M^{-1}((n-k)/2) < d_{te}$, $K_a = K_b \neq K_e$ can be achieved.*

Proof: The error correction capability of $RS(n,\ k)$ is to correct $(n-k)/2$ symbols per codeword. We set a channel proximity threshold d_0, above which the two channels are regarded as different, while below we regard the two channels are identical. Due to the property of wireless channel reciprocity, the estimated channels at the transmitter and the receiver would be almost the same, i.e., $d_{tr} \approx 0$. When $d_{tr} \le M^{-1}((n-k)/2)$, we have $M(d_{tr}) \le (n-k)/2$. Thus, the errors at the receiver can be corrected by the RS code, i.e., $K_a = K_b$.

On the other hand, due to the channel uncorrelation property, the estimated channels at the eavesdropper (e.g., more than several wavelengths away from the receiver) and the transmitter would be totally different, i.e., $d_{te} > d_0$. When $M^{-1}((n-k)/2) < d_{te}$, we have $M(d_{te}) > (n-k)/2$. Thus, the errors at the eavesdropper are unable to be corrected by the RS code, i.e., $K_e \neq K_a$. ■

3) Active attacks: The proposed wireless key establishment scheme may also be targeted by active adversaries, who may try to design, interrupt, intercept, block or overwrite the transmit signals to disrupt the legitimate receiver extracting the secret key specified by the transmitter. For example, an active adversary may jam the communication between the transmitter and the receiver so that the receiver may obtain an incorrect signal. Those attacks are not unique to our scheme, and all previous wireless key establishments are vulnerable to them. For example, Eberz et al. [23] propose a practical man-in-the-middle attack, in which an active attacker is able to impersonate both participants by injecting spoofed packets so Alice and Bob agree on a common key which the attacker knows. In our scheme, however, any injection or jamming by the attacker only changes the key the receiver extracts but is unable to sabotage the key at the transmitter, because the key at the transmitter is always specified by the transmitter itself, instead of extracted from the received signal. As a result, those active attacks would cause discrepancies easily detectable by the transmitter and the receiver. Besides, some techniques have been proposed to remove such active attacks. For example, to defend against jamming attacks, researchers have proposed spread spectrum approaches like Frequency Hopping Spread Spectrum (FHSS) and Direct Sequence Spread Spectrum (DSSS) [19], [20]. We can combine those techniques and the proposed key establishment scheme in order to successfully establish a secret key under active attack scenarios.

V. Experimental Evaluation

A. Methodology

Our prototype system consists of a transmitter (Tx), a receiver (Rx), and an eavesdropper (Ex). Each node is a USRP N210 connected to a PC. USRPs are equipped with SBX daughter boards operating in the the 0.4∼4.4 GHz range as transceivers. The software toolkit is GNURadio [21]. We performed the key establishment in a campus building with small offices and assorted furniture, forming a rich multipath environment. As shown in Figure 4, Rx defines the origin O, and Tx is tested in five other locations (L1∼L5). We compare the obtained key and the original key specified by Tx to determine whether the key establishment is successful, and calculate the $success\ rate$ (i.e., the ratio between the number

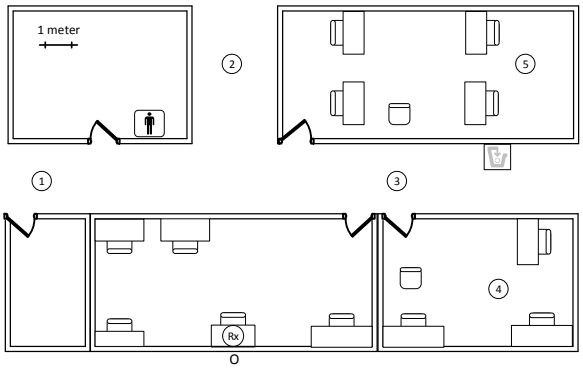

Fig. 4. The floorplan of the key establishment experiment.

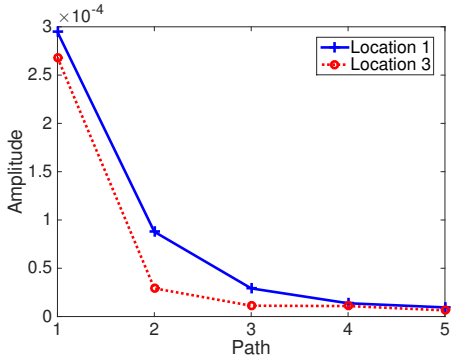

Fig. 5. Channels estimated at two different places.

of successful key establishments and the total number of key establishment trials) through a large range of experiments.

Intuitively, the success rate can be affected by the setting of key mapping (i.e., the path differential), the RS ECC in use for key establishment, the manipulated multipath count, and the key length. In the following experiments, we explore how these factors can impact the key establishment performance.

B. Measuring Channel Proximities

We first verify the spatial uncorrelation property in the experimental environment. As shown in Figure 4, we place Tx at one of the five locations, Ex at another, and Rx at the origin O, repeating the experiment for every unique pair of locations. Both Tx and Ex estimate the channel impulse response between their respective locations and Rx. Note that existing channel estimation algorithms assume a resolvable multipath, and we usually configure the maximum number of resolvable multipaths to an empirical constant value depending on wireless system setups [13]. In this experiment, we consider a channel with five multipath components for our proof-of-concept implementation. We perform 1000 estimates at each location, and thus we obtain 1000 estimation values of the corresponding channel impulse response, denoted by length-5 vectors containing each of the five components. We then calculate the mean of the estimated channel impulse response values for each path in the vector. As an example, Figure 5 shows the mean values of channel estimation results at Locations 1 and 3. The observed channels at two places clearly comprise different shapes, especially in the first three path values.

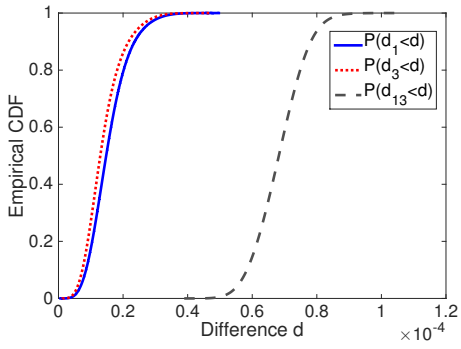

Fig. 6. CDF functions of channel differences.

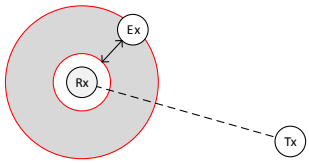

Fig. 7. Setup of eavesdropper (Ex).

Figure 6 plots the empirical cumulative distribution functions (CDFs) of the channel proximities d_1 and d_3 between two channel impulse responses estimated at Locations 1 and 3, respectively, as well as the channel proximity d_{13} between one estimated at L1 and one at L3. We can see that the probability that d_{13} is bigger than d_1 or d_3 is almost 100%. This means we can easily distinguish channels observed at L1 and L3, implying validity of the spatial uncorrelation property of wireless channels. Channel estimation for other pairs of the 5 locations demonstrate similar results and indicate our key establishment scheme should perform well, as we illustrate through the rest of this Evaluation.

We set the central frequency as 1.2GHz. To explore the relationship between the calculated channel proximity at Ex and the distance between Ex and Rx, we change the distance between Rx and Ex every 0.25m, starting from 0.25m (i.e., a wavelength for 1.2GHz signals), and each time we calculate and record the corresponding average channel proximity between the estimated channel at Ex and that at Tx. Table I demonstrates the impact of the distance on the observed channel proximity. We can see with the distance between Ex and Rx increasing, the channel proximity becomes larger, and then maintains a stable high value, which demonstrates that the observed channels at Tx and Ex are uncorrelated.

Besides, we increase the central frequency to 2.4GHz and re-calculate the corresponding channel proximities. A larger central frequency f_c brings a shorter signal wavelength λ (i.e., $\lambda = \frac{3\times10^8}{f_c}$) in theory and therefore decreases the distance required for obtaining an uncorrelated channel. As shown in Table I, for the same distance, Ex observes a higher channel proximity when the central frequency is 2.4GHz, which means the observed channel at Ex in this case has bigger differences with the receiver's channel.

To protect the secret key from Ex, we should keep the observed channels at Ex and Rx uncorrelated. Thus, we keep Ex at least 1m away from Rx, as Table I shows that 1m can guarantee that the observed channel at Ex is uncorrelated with

TABLE I. CHANNEL PROXIMITY ($\times 10^{-4}$) UNDER DIFFERENT f_c

f_c (GHz)	λ (m)	0.25m	0.5m	0.75m	1m
1.2	0.25	0.11	0.35	0.63	0.85
2.4	0.125	0.20	0.51	0.90	0.92

that at Rx. To measure the confidentiality of the generated secret key, we compute and compare the success rates for Ex in breaking the secret key at varying distances away from Rx. As shown in Figure 7, we draw a circle originating at Rx and place Ex at a radius ranging outward from 1 to 4 meters.

C. Path Differential Selection

In the key mapping stage, we set the normalized value of the first path of the manipulated channel impulse response to 0.5, and vary the value of the path differential q from 0.05 to 0.2, with increments of 0.05. For each q, we perform 1000 attempts of establishing a 3-bit key not involving RS ECC between Tx and Rx. Figure 8 shows the success rate of 3-bit key establishment for Rx and Ex when Rx is placed at L1 and Ex is varied from 1 to 4 meters away. We can observe three major tendencies. First, larger distances between Rx and Ex generally result in a lesser ability for Ex to extract the key. When Ex is 1 meter away from Rx, the success rate to break the 3-bit key ranges from 23.7% to 33.6%, while when the distance increases to 4 meters, the success rate falls in the range of 12.8% to 19.3%, which is almost equivalent to the success rate of a random guess (the chance for a random guess to hit the correct 3-bit key is 12.5%). This appears due to the channels for Rx and Ex diverging with increased distance between them. This demonstrates the count of mismatched symbols normally increases with the distance between Rx and Ex, or the observed channel proximity at Ex, i.e., the function $M(\cdot)$ is really monotonically increasing in practice. Second, the success rate at Rx is always much higher that that at Ex. For example, when q value is 0.1, the success rate at Rx and Ex 4 meters away are 83.3% and 14.0% respectively. Finally, the success rate of key establishment increases with the increasing q. When $q = 0.05$, the success rate at Rx decreases to 74.1%, while the success rate at Rx reaches 88.9% when $q = 0.2$. This is because larger q further separates the path responses generated by Tx to better overcome interference.

Though the success rate of establishing a 3-bit key at Rx can reach as high as around 88.9%, this success rate may not be reliable enough to secure private communications in practice. In order to eliminate the bit inconsistence between the specified key bits at Tx and the quantized bits at Rx, we introduce RS ECC to encode the secret key. As $q = 0.1$ achieves the largest difference between the success rate at Rx and that at the surrounding Ex in this initial test for success rate, we employ $q = 0.1$ for the following discussions.

D. RS Code Integration

In this section, we investigate the success rate after incorporating RS code and explore how to select an appropriate RS code. Intuitively, the chosen RS code should make sure that the success rate at Rx can be increased close to 100%, without improving the success rate at Ex and rendering insecure the established key between Tx and Rx.

Our selection criteria for enforced RS code is that its error-correction capability should exceed the $failure\ rate$ (i.e., $1 - success\ rate$) of key establishment at the receiver while remaining below the failure rate at the eavesdropper. Thus, the success rate at the receiver should increase to almost 100% while the success rate at the eavesdropper stays low. As shown in Figure 8, when $q = 0.1$, the failure rate at the receiver is $1 - 83.3\% = 16.7\%$ while the failure rate at the eavesdropper placed at various distances from the receiver is at most $1 - 26.5\% = 73.5\%$. So we utilize RS(7, 3) code which is able to correct 2 symbol errors for a codeword with 7 symbols, which is more than the expected 1.2 symbol errors from a 16.7% failure rate on the receiver but less than the expected 5.1 symbol errors from the eavesdropper's 73.5% failure rate. To further explore the effect of RS code choice on the performance, we enlarge the length of a RS codeword and select another three RS codes under the guidance of the aforementioned selection criteria: RS(15, 7) (4 bits per symbol), RS(31, 17) and RS(31, 15) (5 bits per symbol). For each RS code, we perform 5000 attempts of key establishment using five RS codewords as an example and record the success rate at Rx and the Ex around. Note communicators may establish keys from multiple channel estimations and combine the bits from each to form an aggregate secret key.

Figure 9 plots the success rate of key establishment for Rx, we can observe that the success rate for Rx decreases as the code efficiency (i.e., k/n for RS(n, k)) increases. For example, the code efficiency of RS(7, 3) is 42.9% and its success rate is as high as 97%, while the code efficiency of RS(31, 17) is 54.8% and the corresponding success rate decreases to 78.7%. However, for RS code of higher code efficiency, its encoded key size becomes shorter. A RS(7, 3) codeword can encode a key of length $3 \times 3 = 9$ bits whereas a RS(31, 17) codeword can encode a key of length $17 \times 5 = 85$ bits.

Figure 10 shows the success rate at Ex of different distance away with Rx. Compare with Rx's high success rate (78.7%∼97%), the success rate at Ex is markedly low, ranging from 0.09% to 0.47%. The chosen RS code is unable to correct most errors incurred at Ex, so that while ECC allows the correct decoding of the key at Rx, Ex is unable to use it to decode the key as Rx does. This is because the channels of Ex and Rx are strongly uncorrelated, and the artificial channel impulse response observed by Ex exhibits more bits mismatched with the actual key than Rx. These extra mismatched bits overflow the decoding capability of ECC, thereby leading to the incorrect decoding of the key at Ex.

Choosing the RS code: In general, when the channel of the eavesdropper is uncorrelated with that of the receiver, it is highly likely that a typical RS code would allow the receiver to reconstruct the key with a high success rate and meanwhile yield a very low success rate at the eavesdropper. To further refine the code selection, an empirical profile may be built to reveal the relationship between the number of bit errors and the distance from the receiver. Assuming that the eavesdropper is at least d meters away from the receiver, the communicators can then look up the profile to determine an appropriate ECC code with an error correction capability ranging between x and y, where x is the number of bit errors typically encountered by the receiver and y is the number of bit errors measured d meters away from the receiver.

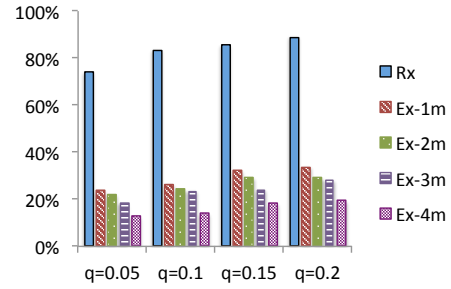

Fig. 8. Success rate vs. value of q.

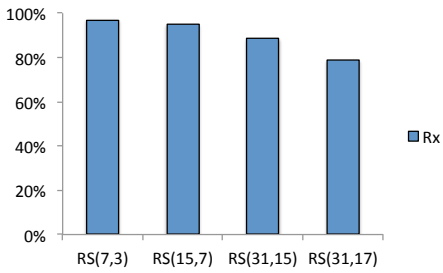

Fig. 9. Success rate vs. RS code type.

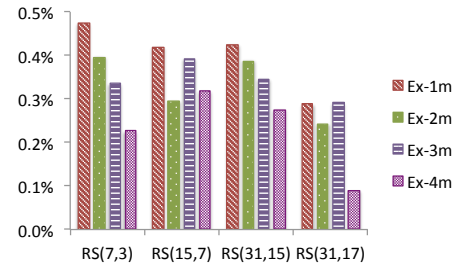

Fig. 10. Success rate for Ex.

TABLE II. SUCCESS RATE (%) VS. KEY LENGTH (BITS)

Key length	Rx	Ex-1m	Ex-2m	Ex-3m	Ex-4m
112	96.0	1.56	1.00	1.21	0.94
140	94.8	0.42	0.29	0.39	0.32
168	91.0	0.10	0.06	0.04	0

TABLE III. SUCCESS RATE (%) VS. VALUE OF L

Value of L	Rx	Ex-1m	Ex-2m	Ex-3m	Ex-4m
3	94.8	0.42	0.29	0.39	0.32
4	92.2	0.34	0.29	0.39	0.30
5	86.3	0.31	0.31	0.41	0.27

E. Key Length

In order to provide effective protection for private communications, the secret key that the transmitter and the receiver intend to establish should be suitably long. Table II shows the relationship between the success rate of key establishment and key length when we utilize RS(15, 7) code to encode the key. This appears as one would expect when varying the length of typical cryptographic keys, demonstrating a clear divergence between success rates for the receiver vs. the eavesdropper with higher key length. Indeed, the success rate for eavesdroppers lowers to 0% with a 168-bit key, but key establishment between legitimate users is still successful.

F. Manipulated Path Count

At the key mapping stage, the transmitter maps a bit sequence with length L_b into a manipulated channel impulse response vector with size L. Correspondingly, the receiver will generate L_b bits as the secret key from an estimated channel impulse response. Previous experiments discuss the situation when $L = 3$, and we now analyze the effect of different L on the performance. Essentially, the number L of manipulated paths in a channel impulse response determines the number of bits that the transmitter can include per channel impulse response. Based on the relationship $L_b = L - 1 + \lfloor L - 1 \rfloor / 2$ mentioned in Section IV-B, when L is set to 4, L_b would be 4, and when L is set to 5, L_b would be 6.

Table III shows the relationship between the success rate of 140-bit key and the number of manipulated paths, using RS(15, 7) code. We see that the success rate at the receiver decreases with the value of L increasing. When $L = 5$, the success rate moves to below 90%; with more paths to manipulate, the key establishment is more susceptible to channel noise. The success rates at the surrounding eavesdroppers, however, do not decrease notably and varies between 0.27% and 0.42%, again demonstrating that the success rate at the eavesdropper mainly depends on the uncertainty of its observed channel.

G. Overall Performance for a Wide Range of Locations

Table IV shows the success rates at the receiver and surrounding eavesdroppers when Tx is placed at each of the locations other than L1 and establishes a 168-bit key utilizing RS(15, 7) code. The number L of manipulated paths ranges from 3 to 5. The success rate at the receiver (ranging from 81.2%∼92.2%) is visibly much higher than those at the eavesdroppers (0%∼0.14%), and successively larger separation causes the eavesdropper's success rate to continue dropping, especially when Ex lies in 4 meters distance away from Rx, it is almost impossible for her to break the key established between Tx and Rx. Through the whole experiment, we show

- Error correction code improves the success rate for Rx while causing no increase in Ex's effectiveness. This follows from the ability to choose an RS code resolving a number of bit errors larger than Rx typically encounters but smaller than the number by Ex.
- Increasing key length to sizes typical of modern keys reduces Ex to near zero efficacy without decreasing the success rate of key establishment.
- Performance of the key establishment method holds steady at all our tested locations and so is concluded to be robust in practice.

VI. RELATED WORK

Since the use of physical layer characteristics of a wireless channel for key generation was first proposed in [22], it has formed a fruitful research area in recent decades, ranging from theoretical analysis to practical experiments [1], [9]. For example, [3] evaluated the effectiveness of secret key extraction from received signal strength variations in wireless channels using real world measurements in a variety of settings.

However, existing wireless key establishments focus on generating keys directly from the channel. They select a certain channel metric and quantize it to obtain bits to form a key. A number of channel metrics have been explored, such as signal envelopes [1], channel impulse response [2], [8], [10], signal phases [11], and received signal strength [3], [7], [12]. In these schemes then, the established key is highly dependent upon the selected channel metric while the communicators themselves hold little control. In our work, we enable the transmitter to specify and control the key at will. Thus, our scheme makes it easier for a transmitter to establish the same secret key with multiple receivers than existing wireless key establishments. For example, the transmitter in

TABLE IV. SUCCESS RATE (%) OF KEY ESTABLISHMENT FOR A WIDE RANGE OF LOCATIONS

	L2			L3			L4			L5		
L	3	4	5	3	4	5	3	4	5	3	4	5
Rx	91.1	87.5	86.0	92.2	87.6	81.2	92.4	89.4	88.1	91.1	85.5	82.9
Ex-1m	0.12	0.12	0.10	0.14	0.10	0.08	0.12	0.08	0.06	0.12	0.10	0.08
Ex-2m	0.10	0.08	0.10	0.12	0.10	0.08	0.10	0.06	0.06	0.14	0.08	0.06
Ex-3m	0.06	0.04	0.02	0.08	0.04	0.04	0.06	0.02	0.02	0.06	0.04	0.02
Ex-4m	0.02	0	0	0	0	0	0	0.02	0	0	0	0

our scheme can act a public key server [23] and distribute a same key to different legitimate receivers. Another significant distinction lies in our work, requiring no reconciliation, where previous efforts [1]–[3], [7]–[11] need reconciliation to correct mismatched bits. [2] points out that assuming Alice and Bob share an authorized channel during this process is unrealistic, leading to the possibility of spoofing attacks. As we do not perform reconciliation, we have no such concerns.

Some other work also utilize manipulated channel information to achieve different goals (e.g., [24], [25]). For example, [24] utilized false channel information to enable the transmitter to hide its location or impersonate another user's location while [25] utilized fake channel information to eavesdrop information received by a target user. In our work, however, we construct a manipulated channel and use it to establish secret keys between legitimate communicators.

VII. CONCLUSION

We propose a novel wireless key establishment technique between a transmitter and receiver pair in the presence of an eavesdropper. Our scheme enables the transmitter to specify any content as the secret key and removes the reconciliation process, which is necessary in conventional wireless key establishments. The transmitter encodes the specified key with error-correction code, maps the encoded key to a channel estimation result, and then utilizes the channel manipulation technique to generate a manipulated signal. On the other end, the receiver uses channel estimation to obtain the encoded key, with potential errors, and then performs error correction to retrieve the secret key. We document real-world implementation on the USRP platform running GNURadio, demonstrating the feasibility and reliability of the proposed technique.

ACKNOWLEDGMENT

This work is supported by the National Science Foundation under grants 1527144 and 1553304.

REFERENCES

[1] B. Azimi-sadjadi, A. Kiayias, A. Mercado, and B. Yener, "Robust key generation from signal envelopes in wireless networks," in *Proc. of ACM CCS*, 2007.

[2] S. Mathur, W. Trappe, N. Mandayam, C. Ye, and A. Reznik, "Radio-telepathy: Extracting a secret key from an unauthenticated wireless channel," in *Proc. of ACM MobiCom*, pp. 128–139, 2008.

[3] 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. of ACM Mobicom*, pp. 321–332, 2009.

[4] A. Sayeed and A. Perrig, "Secure wireless communications: Secret keys through multipath," in *Proc. of IEEE ICASSP*, pp. 3013–3016, 2008.

[5] J. W. Wallace and R. K. Sharma, "Automatic secret keys from reciprocal mimo wireless channels: Measurement and analysis," *IEEE Trans. Information Forensics and Security*, vol. 5, no. 3, pp. 381–392, 2010.

[6] M. Clark, "Robust wireless channel based secret key extraction," in *Proc. of IEEE Milcom*, pp. 1–6, 2012.

[7] 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. of IEEE INFOCOM*, pp. 2283–2291, 2013.

[8] 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. Information Forensics and Security*, vol. 5, no. 2, pp. 240–254, 2010.

[9] Y. Liu, S. C. Draper, and A. M. Sayeed, "Exploiting channel diversity in secret key generation from multipath fading randomness," *IEEE Trans. Information Forensics and Security*, vol. 7, no. 5, pp. 1484–1497, 2012.

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

[11] Q. Wang, H. Su, K. Ren, and K. Kim, "Fast and scalable secret key generation exploiting channel phase randomness in wireless networks," in *Proc. of IEEE INFOCOM*, pp. 1422–1430, 2011.

[12] 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 Computing*, vol. 9, no. 1, pp. 17–30, 2010.

[13] A. Goldsmith, *Wireless Communications*. Cambridge Univ. Press, 2005.

[14] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *Journal of the Society for Industrial and Applied Mathematics*, vol. 8, pp. 300–304, 1960.

[15] IEEE, "IEEE 802.11: Wireless LANs," 2007.

[16] G. D. Durgin and T. S. Rappaport, "Effects of multipath angular spread on the spatial cross-correlation of received voltage envelopes," in *49th IEEE Vehicular Technology Conference*, vol. 2, pp. 996–1000, Jul 1999.

[17] J. Salz and J. H. Winters, "Effect of fading correlation on adaptive arrays in digital mobile radio," *IEEE Transactions on Vehicular Technology*, vol. 43, pp. 1049–1057, Nov 1994.

[18] X. He, H. Dai, W. Shen, and P. Ning, "Is link signature dependable for wireless security?," in *Proc. of IEEE INFOCOM '13*, April 2013.

[19] M. Strasser, C. Pöpper, and S. Čapkun, "Efficient uncoordinated fhss anti-jamming communication," in *Proceedings of the Tenth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, MobiHoc '09, (New Orleans, LA, USA), pp. 207–218, 2009.

[20] S. Fang, Y. Liu, and P. Ning, "Wireless communications under broadband reactive jamming attacks," *IEEE Transactions on Dependable and Secure Computing*, vol. 13, pp. 394–408, May 2016.

[21] "GNU Radio Software." http://gnuradio.org. [Online; accessed in 2014].

[22] J. E. Hershey, A. A. Hassan, and R. Yarlagadda, "Unconventional cryptographic keying variable management," *IEEE Transaction on Communications*, vol. 43, no. 1, pp. 3–6, 1995.

[23] C. Kaufman, R. Perlman, and M. Speciner, *Network security : private communication in a public world*. Prentice Hall series in computer networking and distributed systems, Upper Saddle River (N. J.): Prentice Hall, 2002.

[24] S. Fang, Y. Liu, W. Shen, and H. Zhu, "Where are you from? Confusing location distinction using virtual multipath camouflage," in *Proc. of ACM MobiCom*, pp. 225–236, 2014.

[25] Y.-C. Tung, S. Han, D. Chen, and K. G. Shin, "Vulnerability and protection of channel state information in multiuser MIMO networks," in *Proc. ACM CCS*, (Scottsdale, Arizona, USA), pp. 775–786, 2014.