Dialog Codes for Secure Wireless Communications

Anish Arora and Lifeng Sang
Department of Computer Science
and Engineering
The Ohio State University
Columbus, Ohio 43210
Email: {anish, sangl}@cse.ohio-state.edu

Abstract

We investigate the feasibility of achieving *perfect secrecy* in wireless network communications without shared secrets. We introduce a secure coding problem in which not only the sender but also the receiver participates in the coding. In essence, the receiver's role is to selectively jam the sender's transmission at the level of bits, bytes, or packets. We then design a class of *secure codes*, which we call *dialog codes*, for different channel models and receiver models. Our codes are simple and efficient, with only O(1) complexity in both encoding and decoding process, and achieve optimal coding rate in some channel models. This makes them particularly suitable for low-power resource-constrained devices.

By way of experimental validation, we study the channel jamming characteristics of low-power radios — specifically CC2420 (IEEE 802.15.4) and CC1000— in indoor experiments, observe their time-varying channel behavior, and demonstrate the robustness of our byte-level jamming and packet-level jamming implementations of our dialog codes with respect to dynamic channel fluctuations.

I. Introduction

Message confidentiality is critical to wireless communications, especially in sensitive applications. The conventional approach to message confidentiality relies on using secrets. However, key-based approaches pose several challenges: First, the ease of eavesdropping given the broadcast nature of the medium allows a powerful adversary to easily compromise the secrets and launch further attacks. Second, cryptography with even symmetric secrets can consume significant overhead in wireless networks, especially low power ones. Last but not least, the potentially large number and dynamic nature of nodes pose a difficult key management problem [9].

These challenges lead us to investigate the feasibility of crypto-free secure communication in wireless networks. Crypto-free secure communication has received information-theoretic attention. The authors [6] investigate the possibility of using noisy feedback to achieve secrecy by exploiting the structure of the wiretap channel and using a private key known only to the destination. While they primarily explore the problem from an information theory perspective, we focus on designing dialog codes for different channel model and receiver model.

In order to achieve secure communications without shared secrets, the designated receiver intentionally injects noisy feedback during the sender's transmission, such that any message received at the eavesdropper (if any) has no clue about the source message from the sender. We introduce a secure coding problem as follows: given an m-*bit* input x, the sender encodes x to an n-*bit* message y, and sends y to the receiver. The designated receiver randomly chooses t bits in y to corrupt, such that he is able to recover x from the received message with t corrupted bits by knowing their positions. However, the eavesdropper is completely unknown about x when he overhears the corrupted message without being aware of the corrupted positions, even he knows the coding scheme completely.

Given the above coding problem, we explore guidelines and constraints for any potential solution that achieves *perfect secrecy* (referred to Definition 3). In particular, we show that (1) the low bound for t is m in any model; and (2) the coding rate, $\frac{m}{n}$, is at most 0.5 under a flipping model. Then, we propose a class of dialog codes targeting different channel model and receiver model. Our dialog codes enable lightweight crypto-free wireless communications. They are simple and efficient, with $O(1)$ complexity in both encoding and decoding process. They work under a general channel model regardless of channel fluctuation. We demonstrate the robustness of our byte-level jamming and packet-level jamming implementations of the dialog codes on both CC2420 (*IEEE 802.15.4*) and CC1000 sensor motes. All the source code are available online (see Section V).

Our dialog codes based communication offers several benefits: (1) a large amount of overhead incurred by cryptography operations and key management protocols is saved by the network, hence any to any secure communication becomes possible. This is quite attractive, especially in large wireless sensor networks; (2) Attacks created by compromised secrets are not possible. Compromising one node does not violate the security of others' communications. Our work has revealed a new encryption paradigm that exploits the coding techniques to achieve perfect secrecy. To the best of our knowledge, we are the first to define the secure coding problem in wireless networks, and propose realistic efficient schemes for the coding problem.

The rest of this paper is organized as follows. We pose the system model in Section II, and the secure coding problem and its related properties in Section III. In Section IV, we design a series of dialog codes that achieve *perfect secrecy* and discuss how they can be applied in real applications. In Section V, we experimentally show the channel condition using low power wireless devices and demonstrate the effectiveness of the dialog codes. Section VI reviews related work on crypto-free communications and related security properties. We make concluding remarks and discuss future work in Section VII.

II. System Model

The system consists of a network of either static or mobile wireless nodes. Nodes communicate with their peers in the network or with one or more base station nodes, in case the network has any base stations. Note that we do not require the presence of base stations.

When a sender transmits a message, the designated receiver intentionally injects noisy feedbacks, such that the receiver is able to recover the message with the knowledge of corrupted positions, while the eavesdropper can not infer anything from the corrupted message.

In order to achieve *perfect secrecy* without shared secrets, we assume the following system property:

- **Synchronization**: nodes communications are synchronized. Any bit sent by a node may be corrupted intentionally by a peer.
- **Transmission coverage**: the receiver's transmission power for sending noisy feedback is tuned such that it covers the sender's transmission range.
- **No shared secret**: The coding scheme and related techniques, and channel condition are known to everyone including the eavesdropper.

A. Channel Model

We study a general probabilistic channel model, denoted by **CH**, as shown in Table I. In **CH**, a bit in the source message may be turned into 0 or 1 non-deterministically upon corruption. p is the probability of corrupting 0 to 1, and q is the probability of corrupting 1 to 0. Stability of p and q is not necessarily required in our case, i.e., they may change over time. If $p = q$, this model becomes the well known Binary Symmetric Channel (BSC). If $p = 1$ and $q = 1$, then it becomes a simple flipping model where a bit is flipped deterministically upon corruption.

Original Value	Becomes: 0	Becomes: 1
0	$1-p$	p
1	q	$1-q$

TABLE I

A GENERAL CHANNEL MODEL WHERE THE TRANSITION IS PROBABILISTIC. p AND q SATISFY $p > 0$ AND $q > 0$.

B. Receiver Model

We consider two receiver models as follows, corresponding to a full duplex and half duplex receiver channel respectively:

RC-I: The receiver knows the original value of corrupted bits.

RC-II: The receiver does not know the original value of corrupted bits.

Obviously, model **RC-II** is more general, and of more interest.

C. Threat Model

Because we mainly focus on designing dialog codes such that no one else can discover the message, the adversary in this particular study is simply a passive eavesdropper. It has unlimited computational power. It may physically operate via devices other than the wireless nodes or it may operate via nodes that it compromises. If a node is compromised, its state becomes known to the adversary. In Section IV, we will extend the threat model to see how the secure codes perform if the adversary has more knowledge about the jamming position and value from R. We will also discuss later how to relax the system and the adversary communication assumptions.

III. The Secure Coding Problem

A. Problem Formulation

S	the sender
R	the receiver
E	the eavesdropper
f	a function encoding an m-*bit* input to an n-*bit* message
f^{-1}	a reverse function that decodes the message
x	an m-*bit* input
y	an n-*bit* message sent from S to R. $y = f_{m\rightarrow n}(x)$;
z_R, z_E	messages received at R and E respectively
x^*	an output inferred by E about x
t	the number of bits that R tries to corrupt on y.
η	coding rate: $\frac{size\ of\ the\ source\ message}{size\ of\ the\ encoded\ message}$. $\eta \leq 1$.
$C(w, v)$	the number of combinations of v objects from a set with w objects

TABLE II
NOTATION.

We use the notations listed in Table II throughout the paper.

Our dialog codes are essentially two functions: f and f^{-1}, such that $f_{m\rightarrow n}(x) = y$, $f^{-1}(z_R) = x$, and $f^{-1}(z_E)$ could be any m-*bit* value each with probability $\frac{1}{2^m}$.

Definition 1: A coding scheme has *weak recovery* property if x', recovered from z_R, equals to x by knowing corrupted positions.

The *weak recovery* property in this paper is different from the well known recovery property in the erasure codes, in which the missing (corrupted) positions are generally unknown.

Definition 2: A coding scheme has *randomness* property if x^*, inferred from z_E, could be any m-*bit* value with equal probability $\frac{1}{2^m}$, without knowing the positions of corrupted bits.

The *randomness* property ensures that a coding scheme does not reveal any knowledge about x without being aware of corrupted positions.

Definition 3: A coding scheme achieves *perfect secrecy* if it has both *weak recovery* property and *randomness* property.

Our goal is to design coding schemes that enable secure wireless communication without shared secrets in the presence of eavesdroppers. The coding schemes should suffice to achieve both *weak recovery* property and *randomness* property.

Consider the trivial protocol, as shown in Figure 1:

$$S \rightarrow R: \quad y$$

where S is the sender, R is the receiver, and y is a message. During the transmission of y, R intentionally corrupts some bits in y. The coding scheme should suffice (i) for R to recover the input x based on the knowledge of corrupted positions; (ii) for E to learn nothing at all upon receiving the corrupted message z_E. Note that property (i) implies *weak recovery*; and (ii) implies *randomness*.

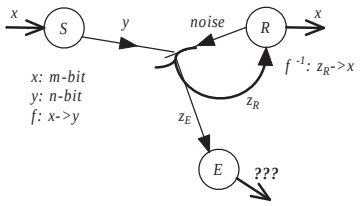

Fig. 1. The communication pattern. *S:* the sender; *R:* the receiver; *E:* the eavesdropper.

In essence, the problem is to find a function, $f_{(m\to n)}(x)$, expanding an m-*bit* input x to an n-*bit* ($n \geq m$) message y, which enables the following actions,

1) The sender S: given any data x, S computes $y = f_{(m\to n)}(x)$, and sends y to R.
2) The receiver R: R tries to corrupt t ($t \leq n$) bits by intentionally injecting noise when S is transmitting y. R receives a corrupted message z_R and knows all the corrupted positions. There exists a function f^{-1}, a reverse function of f, which enables R to discover x by computing $f^{-1}(z_R) = x$ with the knowledge of corrupted positions.
3) The eavesdropper E: E has complete knowledge of the channel situation, the coding scheme f and f^{-1}, the number of noise injections t etc. The only thing it does not know is corrupted positions. f and f^{-1} reveal nothing about x when E receives z_E. In other words, the probability for any instance of an m-*bit* input to generate z_E is $\frac{1}{2^m}$ from E's point of view. Therefore, E can do nothing better than randomly guessing in the space of size 2^m.

A desired coding scheme should have the following properties:

- *Lightweight.* Node processing and communication overhead in realizing the coding scheme is low. By the same token, the latency introduced is low.
- *Efficiency.* Node processing and communication overhead in realizing the coding scheme scales efficiently as the number of input bits m grows.
- *Compatibility.* The coding scheme is easily incorporated into the network stack, i.e., it depends minimally (if at all) on particular network protocols. By the same token, it does not prohibit other security services (e.g. based on cryptography) from coexisting.
- *Availability.* The coding scheme should not be vulnerable to change of channel condition.

B. Property of Secure Coding Scheme

In this subsection, we investigate the general properties of any secure coding scheme that achieves *perfect secrecy* regarding different channel model and receiver model. These properties provide a guideline in designing efficient secure coding schemes. We relegate the design of our dialog codes to the next section.

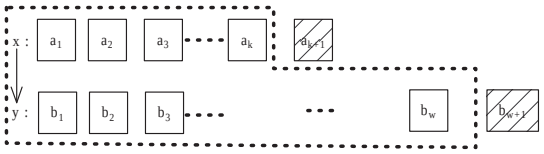

Fig. 2. A new field a_{k+1} (shaded) is added into x when m is increased by 1. At least one new field b_{w+1} should be added into y to reflect a_{w+1}.

Theorem 3.1: A coding scheme that provides *perfect secrecy* must have $t \geq m$.

Proof: We prove this theorem by induction. (I) when $m = 1$, $t \geq 1$ is trivially true since R needs at least one corruption to confuse E; (II) when $m = 2$, we prove $m = 1$ is not sufficient. Because x has 2 bits, there must be 4 different y, each corresponding to an instance of x. In order to provide randomness, any corrupted message must be able to provide the other three variants of x, which is impossible because one corruption can at most bring two confusions, thus $t \geq 2$; (III) suppose when $m = k, k \geq 2$, $n = w$ and $t \geq k$. Now we look at the case of

$m = k + 1$. As shown in Figure 2, at least one new field b_{w+1} should be added to reflect the addition of a_{k+1}. We prove $t = k$ is not enough to provide randomness in this case. Without loss of generality, we assume that at least one corruption on some $b_i, i > w$ would be applied to confuse E about a_{k+1}. If t ($t = k$) corruptions are able to produce 2^{k+1} variants about x, similar to the arguments in (II), the remaining corruptions (at most $k - 1$) on $b_j, 1 \leq j \leq w$ for any instance of x should be able to produce 2^k variants to achieve randomness. This is a contradiction, hence t must be at least m to satisfy the *randomness* property.

Theorem 3.2: Any coding scheme that achieves *perfect secrecy* must have $n > m$ under either (1) receiver model **RC-II**; or (2) receiver model **RC-I** but p and q are not both $\frac{1}{2}$.

Proof: Since an m-*bit* input has a space of size 2^m, which can not be represented by less than m bits, n must be at least m. Next, we prove $n \neq m$. It is trivially true under receiver model **RC-II**, otherwise, R is not able to recover x from an incomplete message z_R. In model **RC-I**, R does not have a recovery problem. Because $t \geq m$ according to Theorem 3.1, every bit in z_E has been corrupted with probability p or q. Because p and q are not both $\frac{1}{2}$, there must be an instance generated with probability greater (or less) than $\frac{1}{2^m}$ when $n = m$. This is a violation to the *randomness* requirement. Theorem 3.2 is proved.

Theorem 3.3: A coding scheme that provides *perfect secrecy* must have $n \geq 2m$ if the channel follows a flipping model under **RC-I**.

In the flipping model, corruption is guaranteed. If E tries to guess x from z_E directly, we have $t \geq m$ according to Theorem 3.1. If E's strategy is to flip every bit in z_E before guessing, then we have to make sure that $\overline{z_E}$ can produce 2^m confusions too. Because z_E is produced by t corruptions on y, $\overline{z_E}$ corresponds to an instance with $n - t$ corruptions on y. Therefore, it is equal to say $n - t$ corruptions must produce 2^m confusions. According to Theorem 3.1, $n - t \geq m$. so $n \geq t + m \geq 2m$. This implies that the optimal coding rate is 0.5 in the flipping channel under **RC-I**.

Theorem 3.4: A coding scheme that provides *perfect secrecy* must have $n \geq 2m$ under **RC-II**.

Recall that *perfect secrecy* implies *weak recovery* and *randomness*. Regarding *weak recovery*, R must be able to recover m-*bit* x from $n - t$ clean bits z_R; Regarding *randomness*, an n-*bit* z_E should produce at least 2^m variants about x each with equal probability $\frac{1}{2^m}$. According to Theorem 3.1, $t \geq m$, so $n \geq t + m \geq 2 \times m$. This theorem implies that the optimal coding rate under **RC-II** is 0.5.

Theorem 3.5: Let x be a concatenation of multiple disjoint parts, $x = A_1 \cdot A_2 \cdot A_k$. If $A_i (1 \leq i \leq k)$ is encoded separately, data dependence among A_i does not help to increase uncertainty.

Proof: Because A_i is encoded separately, z_E can be divided into $B_1, B_2, \cdots, B_k$, each of which corresponds to $A_i, 1 \leq i \leq k$. Let L_i denote the length of decoded message from B_i, the probability of guessing A_i correctly is $p(B_i) = \frac{1}{2^{L_i}}$ by the *randomness* requirement. Therefore, $p(B_j|B_i) \times p(B_i) = p(B_j, B_i) = \frac{1}{2^{L_i + L_j}} = p(B_j) \times p(B_i)$ ($1 \leq i, j \leq k$ and $i \neq j$). This means the conditional probability of B_j given B_i is the same as its own marginal probability, which implies that uncertainty on B_i does help to increase uncertainty on B_j, hence data dependence does not improve uncertainty.

C. How Existing Techniques Perform

Before we present the dialog codes design, we first review existing techniques that may potentially solve our secure coding problem. In order to fit the solutions given by fountain codes (such as LT codes [7], Raptor codes [12]) and secret sharing [11], the secure coding problem is restated at per *symbol* (a *symbol* could be a bit, a byte, or a packet) level as follows,

Given k source symbols $x = \{x_1, \cdots, x_k\}$, $\mathcal{S}$ takes x as input, and produces $n, n > k$ symbols, such that R is able to recover x given any (at least) k from the n symbols.

Here $\mathcal{S}$ is the encoder, which could be LT codes [7], or Raptor codes [12], or the solution for the secret sharing problem presented in [11].

Again, R must corrupt a few symbols to confuse E. In order to minimize the chance of E discovering x, R would better corrupt $n - k$ symbols and recover x from the remaining k symbols. Now we discuss the coding rate for $\mathcal{S}$.

To simplify the analysis, let us assume each source symbol x_i occupy l bits. Assuming the corruption by r be deterministic, E's chance to discover x is at least $\frac{1}{C(n,k)}$. In order to assure perfect secrecy, we must have

$$\frac{1}{C(n,k)} \leq \frac{1}{2^{l \times k}} \tag{1}$$

We first discuss how the coding rate η ($\eta = \frac{k}{n}$ in this case) changes over l with fixed k. Since $C(n,k) = C(n+1,k) \times \frac{n+1-k}{n+1}$, we have $C_k^n < C_k^{n+1}$. If l increases, the right part in Inequation 1 decreases, thus n must increase. As a result, η becomes lower with increasing l. This indicates that η will be quite low when l is fairly large with fixed k.

Next, we discuss how η changes over k with fixed l. According to **Stirling's approximation** [5], we have $n! = \sqrt{2\pi n}(\frac{n}{e})^n(1+\Theta(\frac{1}{n}))$. Using this approximation, Inequation 1 is further derived to be

$$\frac{1}{\eta}log(\frac{1}{\eta}) - (\frac{1}{\eta}-1)log(\frac{1}{\eta}-1) - \frac{1}{2k}log(2\pi(\frac{1}{\eta}-1) \geq l \times log2 \tag{2}$$

This implies that η increases when k increases. However, the upper bound of η must satisfy

$$|\frac{1}{\eta}log(\frac{1}{\eta}) - (\frac{1}{\eta}-1)log(\frac{1}{\eta}-1) - l \times log2| < \epsilon \tag{3}$$

Here ϵ is a small value close to 0. When l is relatively large, η has to be very small, thus $log(\frac{1}{\eta}) \approx log(\frac{1}{\eta}-1)$. Although we can not infer a closed form for the upper bound of η, we know it must be close to $\frac{1}{2^l}$ from Inequation 3. And obviously, it is a very low number. For example, a symbol with size of 16 bits, the coding rate η would be around $\frac{1}{2^{16}}$ to ensure perfect secrecy. For a relatively low l, the upper bound of η is approximated in Figure 3. We see that the best coding rate we may achieve is 0.5 when $l = 1$. This means we should target a coding scheme on the bit level to minimize the coding overhead.

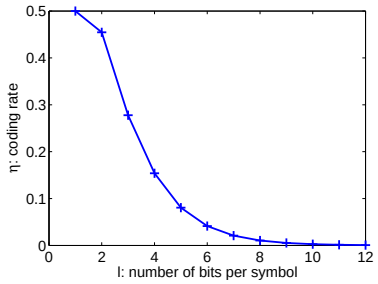

Fig. 3. Upper bound of η with different l.

However, existing techniques such as LT codes and Raptor codes favor large symbols (usually a packet instead of a bit) to make them cost efficient, since each encoded symbol also brings extra information including indexes of all the involved source symbols. They usually target on packet level (e.g. thousands of bits). Otherwise, it is very inefficient if the auxiliary information dominates the information about source symbols in each encoded symbol. Therefore, these existing approaches tend to have a very low coding rate in order to satisfy the perfect secrecy requirement. Even regardless of additional overhead embedded in each encoded packet, a small packet of size 64 bits, for instance, would result in a very low coding rate (around $\frac{1}{2^{64}}$). This motivates us to seek new approaches which provide better coding rate.

IV. Dialog Codes Design

In this section, we describe the design of dialog codes for both the receiver models.

A. Dialog Codes for RC-I

In the receiver model **RC-I**, *weak recovery* is easy to achieve since R knows the original value of corrupted bits. All we need to do is to provide enough *randomness*. We discuss the solutions starting from some special cases considering different p and q:

1) Solutions for Special Cases:

- **Case 1**: $p = q = \frac{1}{2}$. A trivial solution is able to achieve *perfect secrecy*. Let $y = x$ and let $t = m$, the chance for every bit received at E being flipped is $\frac{1}{2}$ when $p = q = \frac{1}{2}$, which means any bit in x could either be 0 or 1 equally from E's point of view. Because all m bits in x are independent, the probability of z_E being produced by any instance of an m-*bit* value is $\frac{1}{2^m}$. *perfect secrecy* is achieved. Note that the channel must be stable all the time to preserve its correctness.
- **Case 2**: $p = q = 1$. According to Theorem 3.3, $n \geq 2m$ in this case. A simple solution in the following can achieve the optimal coding rate 0.5. Let each bit be encoded as a pair of bits, in which the first one is the encoding bit, and the second is a dummy bit. Now R randomly chooses either the first or the second with probability $\frac{1}{2}$ to corrupt. Since the probability of flipping the first bit is half, the probability for E to discover x is $\frac{1}{2^m}$. *Perfect secrecy* is achieved.
- **Case 3**: $p = q$ and $0.5 < p < 1$. In this case, we assume R is so powerful that he knows p instantly. The coding scheme is the same as the one in **Case 2**, but the probability of corrupting the first bit is different. Because R knows p instantly, he corrupts the first bit with probability r such that $r \times p = \frac{1}{2}$. Obviously, such an r exists because $0.5 < p < 1$. If R is not able to discover p instantly, we leave the solution to the general one.

2) A General Solution for RC-I: According to Theorem 3.5, data dependence among different parts does not help increase uncertainty, and our solution is designed for any size of x including one single bit, so we encode each bit independently.

Let v be the current encoding bit, we introduce a sequence of bits $a = \{a_1, a_2, \cdots, a_{k+1}\}$ to represent v, where $a_i (1 \leq i \leq k)$ are random bits chosen by S and $a_{k+1} = v$. We will discuss later how large k is needed to provide *perfect secrecy* with high probability.

Let a be encoded to $b = \{b_1, b_2, \cdots, b_{k+1}\}$ following the rule in Table III. Each bit is encoded based on both the current bit and the previous bit. An input 010, for instance, is encoded as 100 if we assume the very first bit is 0. R is trying to corrupt every bit S sends.

a_{i-1}	$a_i = 0$	$a_i = 1$
0	$b_i = 1$	$b_i = 0$
1	$b_i = 0$	$b_i = 1$

TABLE III

ENCODING SCHEME FOR **RC-I**. EACH BIT IS ENCODED BASED ON ITS OWN VALUE AND ITS PREVIOUS BIT.

Next we prove that E's chance to discover v correctly is close to half by knowing the previous bit at the very beginning. We temporarily consider the case of $p = q$ for ease of understanding the proof.

Let $a^* = \{a_1^*, a_2^*, \cdots, a_{k+1}^*\}$ denote the message inferred by E about a. Let $Pr(a_{i-1}^* = 0) = l_{i-1}$ at E. Without loss of generality, let $a_i = 0$ sent by S, hence the probability for E to discover a_i is as follows,

$Pr(a_{i-1}^* = 0, a_i^* = 0) = l_{i-1} \times (1 - p)$,
$Pr(a_{i-1}^* = 0, a_i^* = 1) = l_{i-1} \times p$,
$Pr(a_{i-1}^* = 1, a_i^* = 0) = (1 - l_{i-1}) \times p$,
$Pr(a_{i-1}^* = 1, a_i^* = 1) = (1 - l_{i-1}) \times (1 - p)$

Therefore,

$$l_i = Pr(a_i^* = 0) = (1 - p) \times l_{i-1} + p \times (1 - l_{i-1}) \tag{4}$$

Let $u_i = l_i - \frac{1}{2}$, Equation 4 becomes $u_i = u_{i-1}(1 - 2p) = u_0(1 - 2p)^i$. Let $l_0 = 1$ known to everyone, so $u_i = \frac{1}{2}(1 - 2p)^i$, which converges to 0 if $p < 0.5$. This implies that l_i converges to half.

Of course, the length of k is dependent on p to achieve *perfect secrecy* with high probability. As an example, if $p = 0.45$, $k = 1.7$ makes $|l_{k+1} - 0.5| < 0.001$.

This scheme also works for $0.5 < p < 1$, because the effect of p is the same as the effect of $1 - p$ if E flips every bit he receives. In the case of $p \neq q$, k can be chosen conservatively such that $\frac{1}{2}(1 - 2w)^k$ is close to 0, where $w = min(0.5 - |0.5 - p|, 0.5 - |0.5 - q|)$.

The coding rate for this scheme is $\frac{1}{k+1}$, where k is the length of the random prefix. Note that this general solution does not work for the case of $p = q = 1$. If $p = q = 1$, see the solution in Section IV-A.1. In reality, receiver model **RC-I** is much harder to have than **RC-II**, so we are more interested in the secure codes for **RC-II**, described in the next section.

B. Dialog Codes for RC-II

In the receiver model **RC-II**, R does not know the original value of corrupted fields. He needs to recover the input x from the incomplete message z_R.

Before we discuss the dialog codes for the general channel model, let us start with a special case: the flipping model where $p = 1$ and $q = 1$. Ideas for this model helps design codes in the general channel model.

1) A Straightforward Solution for the Flipping Model: In the flipping model, corruption is deterministic. We propose the following solution that is surprisingly simple and efficient.

Let each bit in x be expanded to two bits as follows,

$$y_{2i-1}\ y_{2i} = \begin{cases} 0\ 0 & if\ x_i = 0 \\ 1\ 1 & if\ x_i = 1 \end{cases} \tag{5}$$

As a result, an input 010, for instance, is encoded as 00 11 00. R's strategy is to corrupt either field of each pair. Following the above example, if R corrupts all the first bits, then 00 11 00 becomes 10 01 10 at E. R can recover the input simply from the remaining bit of each pair. Since the corruption is deterministic, what E sees are always either 01 or 10, which may come from either 11 or 00 equally. Therefore, the probability for E to make a correct guess for each pair is $\frac{1}{2}$. Thus, E's chance to discover x correctly is $\frac{1}{2^m}$. *Perfect secrecy* is achieved. According to Theorem 3.1, this scheme achieves the optimal coding rate.

However, this coding scheme is vulnerable to corruption failures. If a corruption on any pair of bits fails, an occurrence of 00 or 11 helps E discover these bits directly. Moreover, p and q may not be both 1, and may change over time. In the next section, we propose an efficient solution for the general channel model.

2) A General Solution for RC-II : The key idea behind this scheme is that each encoded bit relies on 2-*bit* history. Every two bits in x are encoded separately. If m is odd, we simply pad one dummy bit in the last to make the input length even. Let x_{2j-1}, x_{2j} $(1 \le j \le m/2)$ be the current encoding pair of bits, we introduce a sequence of bits $a = \{a_1, a_2, \cdots, a_{k+2}\}$ to represent x_{2j-1}, x_{2j}, where $a_i (1 \le i \le k)$ are random bits chosen by S and $a_{k+1} = x_{2j-1}$ and $a_{k+2} = x_{2j}$. k will be discussed later.

a is encoded to $b = \{b_1, b_2, \cdots, b_{2(k+2)}\}$ following the rule in Table IV. For example, a sequence of 0 1 0 would be encoded to 01 10 11 assuming the very first two bits are 0 0. By this, S creates an output y of size n $(n = (k+2)m)$ for delivery. The coding rate is $\frac{m}{n} = \frac{1}{k+2}$, and coding efficiency is $O(1)$ for each bit.

R randomly chooses either bit of each pair to corrupt. In either case, the remaining clean bit is sufficient for R to recover x. Again, the decoding procedure requires $O(1)$ complexity for each bit.

$a_{i-2}\ a_{i-1}$	$a_i = 0$ $b_{2i-1}\ b_{2i}$	$a_i = 1$ $b_{2i-1}\ b_{2i}$
0 0	0 1	1 0
0 1	1 1	0 0
1 0	1 0	0 1
1 1	0 0	1 1

TABLE IV
ENCODING SCHEME FOR **RC-II**.

The remaining question is whether the probability of guessing each x_{2j-1}, x_{2j} $(1 \le j \le m/2)$ correctly converges to $\frac{1}{4}$. Without loss of generality, let $a_{i-2} = 0$, $a_{i-1} = 0$ and $a_i = 0$ (same conclusion can be drawn when they have other values by the symmetry argument), a_i is encoded as 01 according to the encoding scheme in Table IV. Let $c = (c_1, c_2, \cdots, c_{2(k+2)})$ be the corrupted copy of b received at E, and $a^* = (a_1^*, a_2^*, \cdots, a_{k+2}^*)$ be the message

inferred from c about a. Since R may equally choose either bit to corrupt, the probability of each outcome is as follows,

$$\begin{aligned}
&Pr(c_{2i-1}=0, c_{2i}=1) = 1-(p+q)/2,\\
&Pr(c_{2i-1}=1, c_{2i}=1) = p/2,\\
&Pr(c_{2i-1}=0, c_{2i}=0) = q/2
\end{aligned}$$

Suppose the prior probability of the previous two bits at E is as follows,

$$\begin{aligned}
&Pr(a^*_{i-2}=0, a^*_{i-1}=0) = h_1,\\
&Pr(a^*_{i-2}=0, a^*_{i-1}=1) = h_2,\\
&Pr(a^*_{i-2}=1, a^*_{i-1}=0) = h_3,\\
&Pr(a^*_{i-2}=1, a^*_{i-1}=1) = h_4
\end{aligned}$$

where $h_1+h_2+h_3+h_4=1$, and $0 \le h_1, h_2, h_3, h_4 \le 1$.

Consider the case where E thinks $a^*_{i-2}=0$ and $a^*_{i-1}=0$ first. When E sees 01, he makes a successful guess immediately with probability $h_1 \times (1-\frac{p+q}{2})$; When E sees 11 or 00, he knows that it must be corrupted since no such a mapping exists given the history 0 0 according to the encoding table, so he has to guess on this pair. The probability that he fortunately discovers $a_i=0$ is $h_1 \times (p+q)/4$ in this case. Therefore, the total probability that E knows $a_i=0$ is $h_1 \times (1-\frac{p+q}{4})$. By similar calculation, the probability of each possible outcome at E is given as follows,

$$\begin{aligned}
&Pr(a^*_{i-2}=0, a^*_{i-1}=0, a^*_i=0) = h_1 \times (1-\tfrac{p+q}{4}),\\
&Pr(a^*_{i-2}=0, a^*_{i-1}=0, a^*_i=1) = h_1 \times \tfrac{p+q}{4},\\
&Pr(a^*_{i-2}=0, a^*_{i-1}=1, a^*_i=0) = h_2 \times (\tfrac{1}{2}+\tfrac{p-q}{4}),\\
&Pr(a^*_{i-2}=0, a^*_{i-1}=1, a^*_i=1) = h_2 \times (\tfrac{1}{2}+\tfrac{q-p}{4}),\\
&Pr(a^*_{i-2}=1, a^*_{i-1}=0, a^*_i=0) = h_3 \times \tfrac{p+q}{4},\\
&Pr(a^*_{i-2}=1, a^*_{i-1}=0, a^*_i=1) = h_3 \times (1-\tfrac{p+q}{4}),\\
&Pr(a^*_{i-2}=1, a^*_{i-1}=1, a^*_i=0) = h_4 \times (\tfrac{1}{2}+\tfrac{q-p}{4}),\\
&Pr(a^*_{i-2}=1, a^*_{i-1}=1, a^*_i=1) = h_4 \times (\tfrac{1}{2}+\tfrac{p-q}{4})
\end{aligned}$$

Let the marginal probability of guessing a_i correctly be l_1. Summing up the cases where $a^*_i=0$, we get $l_1 = h_1 \times (1-\frac{p+q}{4}) + h_2 \times (\frac{1}{2}+\frac{p-q}{4}) + h_3 \times \frac{p+q}{4} + h_4 \times (\frac{1}{2}+\frac{q-p}{4})$. Note that l_1 becomes $1/2$ if $h_1=h_2=h_3=h_4=1/4$, and we are done. If h_1, h_2, h_3, and h_4 are not equal, let us look at the marginal probability of new history, a^*_{i-1} and a^*_i, which will be the indicator for the next bit a^*_{i+1}. The normalized marginal probability is given by,

$$\begin{aligned}
&Pr(a^*_{i-1}=0, a^*_i=0) = (h_1+h_3) \times l_1,\\
&Pr(a^*_{i-1}=0, a^*_i=1) = (h_1+h_3) \times (1-l_1),\\
&Pr(a^*_{i-1}=1, a^*_i=0) = (h_2+h_4) \times l_1,\\
&Pr(a^*_{i-1}=1, a^*_i=1) = (h_2+h_4) \times (1-l_1)
\end{aligned}$$

Let $l_0 = h_1+h_3$, then the above equations become

$$\begin{aligned}
&Pr(a^*_{i-1}=0, a^*_i=0) = l_0 \times l_1,\\
&Pr(a^*_{i-1}=0, a^*_i=1) = l_0 \times (1-l_1),\\
&Pr(a^*_{i-1}=1, a^*_i=0) = (1-l_0) \times l_1,\\
&Pr(a^*_{i-1}=1, a^*_i=1) = (1-l_0) \times (1-l_1)
\end{aligned} \tag{6}$$

For simplicity, let us consider the case where $a_{i+j}=0$ for $j \ge 0$. Then the marginal probability can be generalized as

$$\begin{aligned}
&Pr(a^*_{i+j}=0, a^*_{i+j+1}=0) = l_{j+1} \times l_{j+2},\\
&Pr(a^*_{i+j}=0, a^*_{i+j+1}=1) = l_{j+1} \times (1-l_{j+2}),\\
&Pr(a^*_{i+j}=1, a^*_{i+j+1}=0) = (1-l_{j+1}) \times l_{j+2},\\
&Pr(a^*_{i+j}=1, a^*_{i+j+1}=1) = (1-l_{j+1}) \times (1-l_{j+2})
\end{aligned} \tag{7}$$

where

$$\begin{aligned} l_{j+2} &= l_j l_{j+1}(1-\frac{p+q}{4}) + l_j(1-l_{j+1})(\frac{1}{2}+\frac{p-q}{4}) \\ &+ (1-l_j)l_{j+1}\frac{p+q}{4} + (1-l_j)(1-l_{j+1})(\frac{1}{2}+\frac{q-p}{4}) \\ &= (l_{j+1}(1-p)+\frac{p-q}{2})(l_j-1/2)+1/2. \end{aligned} \tag{8}$$

Let $v_j = l_j - 1/2$ where $-1/2 \le v_j \le 1/2$, then the above equation becomes

$$v_{j+2} = v_j \times (v_{j+1}(1-p) + \frac{1-q}{2}) \tag{9}$$

It is easy to check that v_j converges to 0. For example, if $v_0 > 0$, let $w = min(p,q)$, we have $v_{j+2} \le v_j(v_{j+1} + \frac{1}{2})(1-w) \le v_j(1-w) \le max((1-w)^{(j+2)/2} \times v_0, (1-w)^{(j+1)/2} \times v_1)$. When $v_0 < 0$, v_j also converges to 0 by a similar argument. Therefore, $l_j \to 1/2$, and the probability of guessing the next two bits correctly converges to $1/4$. The same conclusion can be drawn with similar arguments above for arbitrary a_i, with slight change in Equation 7. Because the probability of guessing each x_{2j-1}, x_{2j} $(1 \le j \le m/2)$ correctly is $\frac{1}{4}$, the probability of guessing x correctly is $\frac{1}{2^m}$.

The length of k required to confuse E is dependent on the value of p and q. As an example, let $h_1 = 1, h_2 = 0, h_3 = 0, h_4 = 0$ initially, the plot of j versus $w = min(p,q)$ such that $|l_j - 0.5| < 0.001$ is illustrated in Figure 4. For example, 14 bits are roughly enough to confuse E when $w = 0.3$. If w is relatively large, then k can

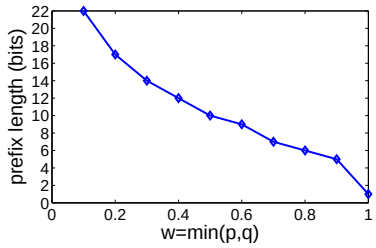

Fig. 4. Number of bits required to create uncertainty. $|l_j - 0.5| < 0.001$.

be relatively short. If $p = q = 1$, the scheme achieves the optimal coding rate 0.5. Note that this scheme is not affected by changing p and q over time if k is chosen conservatively.

Based on the above discussion, we know that the uncertainty of the received messages at E comes from the absence of knowledge about the location R chooses to corrupt and the channel uncertainty. We ask the question: will dialog codes be broken if E happens to know what and where R injects? We discuss this extension of the threat model with a detailed example. Let the probability of R injecting 0 be α, and the probability of R choosing the first position to corrupt be β. Let $Pr(0\ 0 \to 0) = d_1$, $Pr(0\ 1 \to 0) = d_2$, $Pr(1\ 0 \to 1) = d_3$, $Pr(1\ 1 \to 1) = d_4$, where the pattern $Pr(a\ b \to c)$ means the probability of E receiving c when S sends a while R corrupts with b. Without loss of generality, let S send 0 1 (codes for input 0), the probability of E receiving 0 1 correctly is as follows,

$$Pr(Z_e = 0\ 1) = \beta(\alpha d_1 + (1-\alpha)d_2) + (1-\beta)(\alpha d_3 + (1-\alpha)d_4) \tag{10}$$

Even if the adversary knows α, β, and/or $d_i, i = 1,2,3,4$, $Pr(Z_e = 0\ 1)$ is always less than 1. In other words, E will finally get lost by the cumulative uncertainty produced in the prefix. In the extreme case, when the adversary knows the exact position and value that R will jam, $\beta = 1, \alpha = 1$ for example, then $Pr(Z_e = 0\ 1) = d_1$, which is the best that E can achieve. Again, the dialog codes will not be broken if the prefix is chosen conservatively. In the next section, we will see that it is very hard to discover d_i (or p and q), not mentioning α and β.

V. Experimental Study

In this section, we study the channel condition induced by both *CC1000* motes and *CC2420* motes [8], and explore the feasibility of implementing the dialog codes on sensor network platforms. Each $CC1000$ mote consists of a $4MHz$ ATmega128L microcontroller, $128KB$ of flash and $4KB$ of RAM; the radio frequency is $433MHz$. Each $CC2420$ mote consists of a $8MHz$ MSP430 microcontroller, and operates at $2.4GHz$. It has $48KB$ of flash and $10KB$ of RAM. We use TinyOS [1] in our implementations. We use channel 2 for $CC1000$ motes and channel 15 for $CC2420$ motes in all the following experiments.

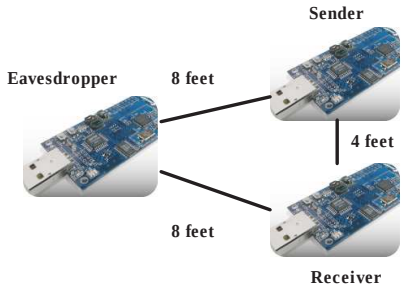

Fig. 5. The setup of the experiments. The sender and receiver are 4 feet away, while the eavesdropper is 8 feet away from both the sender and the receiver.

In order to enable accurate evaluation, we have to take care of the following implementation issues: (I) In the default implementation in TinyOS, A transmission is only enabled when the channel is free. The default CSMA protocol backoff if the channel is busy to avoid collision and interference. However, our case must allow R to inject noise signal no matter whether the channel is busy or not; (II) In the default implementation, when a message is ready to send, multiple tasks, from higher layers to lower layers, will be posted to take the message buffer, write the payload to the data registers, write command on transmission registers, and send the message over the radio stack. Similarly, when a message is received, multiple tasks are issued to process the message and signal events to inform higher layers. Because tasks in TinyOS are a form of deferred procedure call (DPC) [4], which enable a program to defer a computation or operation until a later time, such an implementation in TinyOS may introduce unexpected delays that may degrade the accuracy of our evaluation.

The above considerations lead us to disable the default CSMA protocol at R so that its intentional noise can be sent immediately. In addition, in order to remove processing latency introduced by the task mechanism, we go through the two MAC/radio stacks (each for *CC1000* motes and *CC2420* motes), and modify the TinyOS library such that a *send* command and a *receive* event are executed immediately without involving tasks. In cases where the transmitting payload is the same, we may directly call the transmission register to issue a *send* command without reloading the data.

A. Studying Channel Condition

In the study of channel condition, we are particularly interested in what p and q in the channel model could be in real scenarios.

Three motes, representing the sender, the receiver and the eavesdropper respectively, are used in our experiments. In order to achieve fairly accurate results, we further take care of the following issues: (1) Communication latency. It is often thought that signal travels so quickly that the time taken for it to get from its source to its target would be irrelevant. However, to minimize the potential deviation of communication latency caused by different distance between (S, E) and (R, E), we deploy the motes such that the eavesdropper is 8 feet away from both the sender and the receiver. The setup is shown in Figure 5. (2) Transmission power. Different transmission power may result in different collision and interference. In our case, each mote has the same transmission power level, which is tuned such that all three motes are within the reliable communication range from each other. (3) Synchronization. To intentionally create collision, we need to synchronize the transmissions. We let E to trigger the communication, i.e., both S and R start sending a message immediately after they receive a control message from E. Note that E acts as an auxiliary node rather than an attacker in this particular evaluation. (4) Payload. The payload, as well as

the position of S and R, is switched to avoid sampling deviation introduced by hardware variance. Note that we also overwrite the TinyOS message header with our designated value so that the original header will not affect the results.

1) Fixed Distance and Transmission Power: We first investigate the channel condition with fixed distance among S, R and E using same transmission power level 3. we carry out a series of experiments. In all experiments, E sends a control message every 100 milliseconds, and S and R each send a message (38 bytes including the TinyOS message header) with different payload as follows,

1) S: **all 0's**; R: **all 1's**.
2) S: **all 1's**; R: **all 0's**.
3) S: **all 0's**; R: **all 0's**.
4) S: **all 1's**; R: **all 1's**.
5) Repeat the above 4 experiments after switching the position of S and R.

In each experiment, E records all the messages received and computes the percentage of 0's and 1's respectively. Because the control message is sent every 100 ms, we are able to check whether there is a pair of messages (one from S and the other from R) both received within the 100 ms interval. However, we do not find any such a case in all the experiments. This means that either the messages are collided successfully, or E is not able to take the other message when there is already one in reception due to the transceiver limitation on the device.

	# samples	0	1
S: 0; R: 1	1432640	37.74%	62.26%
S: 1; R: 0	1995840	65.68%	34.32%
S: 0; R: 0	1182720	65.27%	34.73%
S: 1; R: 1	2068480	37.77%	62.23%

TABLE V

EXPERIMENTAL RESULTS FOR THE CHANNEL CONDITION USING *CC1000* MOTES.

A summary of the experiments using *CC1000* motes is presented in Table V. We see that the probability of flipping a bit is fairly high. For example, the chance of producing bit 1 by colliding 0 with 0 is 34.73%, while the chance of producing bit 0 by colliding 1 with 1 is also above 30%. As an example, we use this result see how p and q may look like. Note that p is the probability of turning 0 to 1. We use the data in which S sends all 0's. Since R could either jam 0 or 1, we average the probability for the cases $0\ 1 \rightarrow 1$ and $0\ 0 \rightarrow 1$, and we get $p = 0.435$. Similarly, we get $q = 0.517$. In this particular case, the channel is not symmetric, but p and q are close.

Because of page limitation, we put a thorough study of the channel condition on the CC2420 motes in Appendix. The observations of both motes in this study is concluded as follows,

- The probability of creating ($0\ 0 \rightarrow 1$) and ($1\ 1 \rightarrow 0$) by collision is relatively high on the *CC1000* motes, while it is fairly low on the *CC2420* motes. In contrast, the probability of producing ($1\ 0 \rightarrow 0/1$) and ($0\ 1 \rightarrow 0/1$) is fairly high on both types of motes.
- On average, p and q are reasonably high on both motes, varying around $0.3 - 0.7$.
- Hardware variation and environmental settings (e.g. location of the devices) both affect the outcome by collision.
- The channel condition, or p and q, changes frequently over time.

2) Varying Distance and Transmission Power: In this particular study, we are interested in the following questions: will the eavesdropper benefit from sitting closer to the sender? will the collision be less likely to happen if R lowers its transmission power although R's noise still covers S's transmission range? Here we investigate the impact of varying distance and transmission power. We carry out the following series of experiments,

- In this set of experiments, S, R and E are sitting on a line. The distance between S and E is fixed, 8 feet. R starts from sitting at middle between S and E, 2 feet away from E, and moves away from E with a step 2 feet, as shown in Figure 6. At each location, we carry out one experiment in which S sends all 1's, and R sends all 0's both at transmission power level 3. We record the collision results at E.

- In this set of experiments, the setup for S, R and E is as in Figure 5. S sends all 1's at power level 6, and R sends all 0's at varying power level from 3 to 9. We record the collision results at E.

The varying-distance result is shown at Figure 7, and the varying-power result is shown at Figure 8. We see that R's signal does not necessarily dominate S's signal by either decreasing distance between R and E, or increasing transmission power at R. This means (1) E does not necessarily benefit from sitting closer to the sender than R; (2) E does not necessarily benefit from R's lowering its transmission power (without violating that S and R are within reliable transmission range).

Fig. 6. Experiment setup for varying distance between E and R.

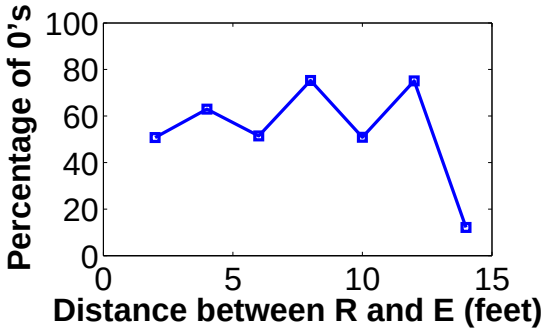

Fig. 7. Varying distance between E and R.

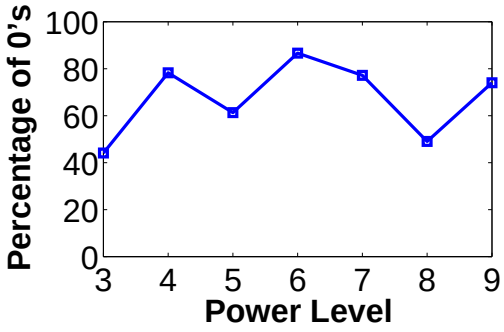

Fig. 8. Varying R's transmission power level.

Note that these observations are limited to the Tmote Sky device. Different devices may have different characteristics. The purpose of these studies is to verify the correctness of our channel assumption, i.e., $p > 0$ and $q > 0$. On the other hand, our empirical observations may be beneficial in the future design and other research as well. The source code of this study is available online (the link is removed for blind review).

B. Implementation of Dialog Codes

Currently, we are not able to implement the dialog codes on the hardware level. However, we have implemented a byte-level version on *CC1000* motes and a packet-level version on *CC2420* motes using TinyOS to demonstrate the effectiveness of the dialog codes. This is the lowest level we can manage a message due to hardware limitation. We use a byte to represent a bit: the XOR value of all the bits in the message represent a bit in the dialog codes. Although this is the best we can do currently, this implementation can be improved in real applications if they are targeted on the hardware level. The only purpose of this simulated implementation is to validate the correctness of the dialog codes. We leave the real implementation as our future work.

As a demonstration, we use the same setting as that in the channel study. The experiment is performed as follows:

(1) Before transmission, S encodes the message following the scheme in Table IV. Each bit is then further extended to a random message (fixed size) such that the XOR of all the bits in the message is equal to the encoded bit.

(2) After encoding, S starts sending messages. Each encoded byte is preceded by two bytes for synchronization purpose, so that R knows when to inject noise. R randomly injects a byte once for every next pair of bytes upon receiving the sync bytes, and records the position he corrupts and remaining byes he receives as well. E ignores the sync bytes, and records all other information.

By this, R is able to correctly decode the message, while E does not have any information about the input. Here are example outputs from multiple tests in which E tries to decode the messages following the mapping table,

Input at S:

0xa1,0xb2,0xc3,0xd4

Sample output after decoding at E:

0xb4 0x28 0xb4 0x21
0x43 0x65 0x87 0xa9
0x0d 0x96 0x1e 0xa5
0x36 0x58 0x7a 0xa8
...

The source code of this simulated implementation is also available online (the link is removed for blind review).

VI. Related Work

Crypto-free wireless secure communication has received information-theoretic attention [6], where the authors explore the utility of user cooperation in facilitating secure wireless communications. Their scheme exploits the structure of the wiretap channel and uses a private key known only to the destination. Our work in this paper complements [6] by introducing a secure coding problem and proposing dialog codes to achieve *perfect secrecy* under a general channel model.

The secret sharing problem [2] [11] is to share a secret x among a set of n participants. The secret is encoded such that the secret is reconstructable from any $k, k < n$ pieces, but even complete knowledge of $k-1$ pieces reveals absolutely no information about x. The main difference between our problem and the secret sharing problem is the threshold behavior. In addition, the receiver is aware of the positions of the corrupted bits in our problem.

The verifiable secret sharing problem [3] lets each player to verify the shared secret from a certain amount of information in which some may be false. A simple version of this problem is as follows: A player receives n pieces of information about the dealer's secret, among which k pieces are false. A verifiable scheme should allow the player to reconstruct the dealer's secret given the n pieces information. In our problem, E knows at most t bits are corrupted, but he does not know which t bits are corrupted. On the other hand, the recovery procedure is easier for R because he has the knowledge about the corrupted positions.

The fountain codes (also known as rateless erasure codes), including LT codes [7] and Raptor codes [12], are a class of erasure codes with the property that a potentially limitless sequence of encoding symbols can be generated from a given set of source symbols such that the original source symbols can be recovered from any subset of the encoding symbols. These techniques may be tailored to be potential solutions for our secure coding problem, however, as discussed in Section III-C, they tend to have a very low coding rate to ensure perfect secrecy, and thus are very inefficient for the security purpose.

In another related work [10], the authors have successfully achieved light-weight crypto-free authentication and non-repudiation using spatial signatures in wireless sensor networks. The dialog codes in this work complements [10] by providing confidentiality for wireless communications.

VII. Discussion and Conclusion

In this paper, we investigate the feasibility of crypto-free wireless communication, and define a secure coding problem to achieve *perfect secrecy* without shared secrets. General properties and constraints are explored to provide guidance for further design of any coding scheme. Then, we propose a series of dialog codes considering different

channel condition and receiver model. Our dialog codes are simple and efficient, achieving $O(1)$ complexity in both encoding and decoding process. They work under a general channel model, and are immune from channel fluctuation over time. Our dialog codes enable lightweight secure wireless communication without shared secrets. We have implemented the dialog codes on both byte level and packet level using low power wireless devices including *CC1000* motes and *CC2420* motes, and demonstrated its effectiveness by in-door experiments. Our dialog codes is not only suitable for regular data encryption, but also useful for key initialization if the upper layer application prefers key based encryption.

So far, we have not considered communication loss and malicious injections. Mechanisms such as checksum that ensures the integrity of a message may be used to deal with these problems, as they do in normal wireless communications. Current dialog codes for the general channel model may not be compact enough from the coding rate perspective. Whether there exists any compact coding scheme for our secure coding problem is still an open problem.

In this paper, we only focus on corruption rather than insertion about the noisy feedback under the assumption of synchronization. Real implementation may have to deal with problems such as insertion. In addition, the coding scheme may need to be slightly adjusted considering the level of synchronization that the devices can achieve in real implementations, as we do in our byte level and packet level implementation. Other future work includes improving the implementation of the dialog codes on small devices at hardware level.

VIII. Acknowledgment

This material is based upon work supported by the National Science Foundation under Grants No. 0520222 and 0341703.

References

[1] http://www.tinyos.net.
[2] G. Blakley. Safeguarding cryptographic keys. *Proc. AFIPS 1979 NCC*, 48:pp. 313–317, June 1979.
[3] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch. Verifiable secret sharing and achieving simultaneity in the presence of faults. *FOCS85*, pages pp. 383–395, 1985.
[4] E. Cota-Robles and J. P. Held. A comparison of windows driver model latency performance on windows nt and windows 98. *In Proceedings of the Third Symposium on Operating System Design and Implementation (OSDI)*.
[5] T. H.Cormen, C. E.Leiserson, R. L.Rivest, and C. Stein. Introduction to algorithms. *The MIT Press*, 2001.
[6] H. E. G. L. Lai and H. V. Poor. The wiretap channel with feedback: Encryption over the channel. *submitted to the IEEE Transactions on Information Theory*, April 2007.
[7] M. Luby. Lt codes. *Foundations of Computer Science, 2002.*, pages 271 – 280, 2002.
[8] MoteIV. http://www.moteiv.com/.
[9] A. Perrig and J.D.Tygar. Secure broadcast communication in wired and wireless networks. *Kluwer Academic Publishers*, 2003.
[10] L. Sang and A. Arora. Spatial signatures for lightweight security in wireless sensor networks. *submitted to Infocom 2007*, 2007.
[11] A. Shamir. How to share a secret. *Communications of the ACM*, 22(11):612 – 613, 1979.
[12] A. Shokrollahi. Raptor codes. *IEEE Transactions on Information Theory*, 52(6):2551 – 2567, 2006.

IX. Appendix

More experimental results can be found http://www.cse.ohio-state.edu/s̃angl/paper/sang_jamming.pdf