Dialog Codes for Secure Wireless Communications

Anish Arora and Lifeng Sang
Department of Computer Science & Engineering
The Ohio State University
{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 diverse channel models and receiver models. Our codes are simple and efficient, with only O(1) complexity in both the encoding and the decoding process, and achieve optimal coding rate in some channel models. This, along with their potential for augmenting security and/or simplifying security bootstrapping, makes them worthy of consideration for resource-constrained wireless sensor network devices.

By way of experimental validation, we study the channel jamming characteristics of extant mote radios — specifically, CC2420 (IEEE 802.15.4) and CC1000— in experiments, observe their time-varying channel behavior, and demonstrate the correctness and robustness of implementations of our dialog codes at the byte-level and at the packet-level in the presence of dynamic channel fluctuations.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks:**]: Security and protection

General Terms

Algorithm, Security

Keywords

Dialog Codes, Security, Wireless Sensor Network

IPSN'09, April 15–18, 2009, San Francisco, California, USA.

1. INTRODUCTION

In recent years, there has been a resurgence of interest in physical layer security in wireless communications. Information-theoretic security principles form the basis for the recent studies. In contrast to the seminal work of Shannon, which largely motivated the study of computational security, the seminal work of Wyner [23] showed that perfect security could be achieved even without the use of shared secrets. While Wyner's work has been built upon in many ways since it was introduced and especially recently, the focus thus far has largely been on theory. The development of practical codes, the study of their impact on security architectures as well as on cross-layer design of security, and their evaluation in current wireless platforms is only just beginning.

In this paper, we take a first step in this direction by designing and evaluating a perfect security code for confidential communications that does not use any shared secrets. To this end, we define a secure coding framework wherein perfect secrecy is achieved in the communications from a sender to a receiver as follows. While the sender is transmitting, the receiver plays an active role in preventing passive eavesdroppers from being able to successfully decode the sender's message.

The challenge then is to ensure that the sender can encode its message so that regardless of which parts are jammed, the receiver is still in a position to recover the sender's message from the information it receives, but the eavesdropper —which is assumed to be incapable of distinguishing the jammed parts from unjammed parts— cannot decode the message any better than it would by random guessing. Note that this last requirement must be satisfied even when the eavesdropper is aware of the functions used for encoding, jamming, and decoding. We will refer to this challenge as the secure coding problem; its precise definition is given in the next section.

Our solution to the secure coding problem is the Dialog Codes. They are derived from a basic strategy as follows. Each source bit is augmented with a redundant bit; the receiver randomly jams either bit in a

pair. Since the eavesdropper does not know which bit is jammed or the output would be, she can not recover the jammed bit or decode the message correctly.

The implications of not using shared-secrets (in either symmetric or asymmetric form) for achieving secure communications are manifold. To begin with, solutions to the secure coding problem at the physical layer can be seen as a way of increasing the security level of wireless networks wherein, say for reasons of resource limitation, a highly secure protocol cannot be implemented at higher layers. By the same token, they can be used as a building block in efficiently bootstrapping the security parameters and configuration data required by higher layer protocols and applications; bootstrapping is widely regarded as being a hard and important problem for deeply embedded and potentially large scale wireless networks. Finally, it is conceivable that in some application scenarios that it is possible to completely eschew the use of shared secret based cryptosystems.

Contributions of the paper. We present the general properties of any solution to the secure coding problem, showing in particular that the coding rate of any solution in the general half-duplex communication model is upper bounded by 0.5. We also design a class of dialog codes, whose instances span different channel models and receiver models. Broadly speaking, our dialog codes enable lightweight wireless communications in the sense that they are simple to implement and efficient to compute, with $O(1)$ time complexity for both their encoding and decoding functions. They withstand the variations of channels, including those of low-power channels the complexity of whose behaviors has been deeply studied in recent wireless sensor network research.

We validate our assumptions about the ability of receivers to predictably jam (and corrupt) communications, via an experimental study on two wireless sensor network platforms, the *CC2420* (*IEEE 802.15.4* compatible) motes and the *CC1000* motes. The study also corroborates the jamming model we use in proving the perfect secrecy of our dialog code designs. We also demonstrate (at the level of individual traces) the correctness and robustness of our byte-level jamming and packet-level jamming implementations of dialog codes, respectively on the *CC1000* and *CC2420* motes. Finally, although we have evidence that the assumption that the eavesdropper cannot detect when jamming is occurring is appropriate for some wireless platforms, we explain how the use of dialog codes is not fundamentally dependent on the assumption.

Road map. The rest of this paper is organized as follows. We define the problem and the system model in Section 2. We then discuss properties of potential solutions in Section 3. In Section 4, we explore the possibility of using existing techniques to solve the secure coding problem, by way of motivating the need for dialog codes. We design various dialog codes in Section 5. In Section 6, we experimentally validate the jamming properties of wireless sensor network platforms and demonstrate the validity of the dialog codes. Finally, we review related work in Section 7, and make concluding remarks as well as discuss future work in Section 8.

For the reader's convenience, we summarize the notations used in the rest of this paper below.

j, k, e	sender, receiver & eavesdropper
x_i	the i^{th} bit in message x
$\mathcal{S}$	domain of private messages
$\mathcal{X}$	domain of messages sent by j
$\mathcal{Y}$	domain of messages received by k
$\mathcal{Z}$	domain of messages overheard by e
Γ	domain of random time sequence
f	encode function f: $\mathcal{S} \rightarrow \mathcal{X}$
ϕ	decode function ϕ: $\mathcal{Y} \times \Gamma \rightarrow \mathcal{S}$
$j \langle\sim_\gamma k$: x	j shouts out a message x, k uses time sequence γ to selectively jam
$p^w_{u\not\triangleleft v}$	probability of producing value w at e when a bit u is jammed by v

Table 1: Notations

2. PROBLEM STATEMENT AND SYSTEM MODEL

Recall the general wire-tap channel shown in Figure 1. In this model, an important result is that information secrecy can be achieved only when the channel at the eavesdropper is somehow degraded relative to that at the receiver [23, 7, 12].

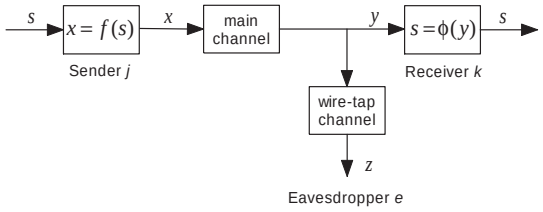

Figure 1: General wire-tap channel

The approach we adopt extends Wyner's wiretap model to ensure degradation at the eavesdropper by letting the receiver cooperatively jam the sender during the sending of a message. To do so, the receiver unilaterally chooses a time sequence at which it jams the sending; it does not reveal the chosen time sequence to other nodes. We illustrate the extended model in Figure 2. Note that the extended model allows for the possibility that in the absence of the cooperative jamming by the receiver the

wiretap channel has better gain than the main channel between the sender and the receiver.

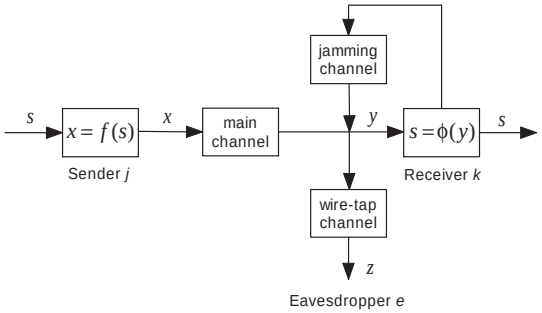

Figure 2: Wire-tap channel with cooperative jammer

Informally speaking, the problem then is to maximize the rate of reliable communication from j to k, subject to the constraint that e learns as little as possible about the source output [23]. More formally:

The Secure Coding Problem. Given an arbitrary message s, $s \in \mathcal{S}$, that node j wishes to send privately to node k and an arbitrary time sequence γ, $\gamma \in \Gamma$, that node k uses to cooperatively jam while j is sending. Design coding functions f and ϕ for $j \langle\sim_\gamma k$: x such that

- $x = f(s)$, where $x \in \mathcal{X}$
- $\phi(y, \gamma) = s$, where $y \in \mathcal{Y}$
- $Pr(s) = Pr(s|z)$, where $z \in \mathcal{Z}$

We will focus on solving the secure coding problem at the level of bits. In other words, our goal is to find f and ϕ, such that for any m-bit source message s, j encodes s via f into an n-bit message x, k decodes s via ϕ and the knowledge of γ, while e guesses s correctly with a probability that not better than $\frac{1}{2^m}$ even if it knows f and ϕ (regarded as perfect secrecy). Note that our secure coding problem simplifies the channel model to be reliable channel and the only errors considered are induced by jamming.

In the rest of this paper, we assume the following system properties:

1. Cooperative jamming by the receiver is predictable in the sense that the probability of j's message being corrupted by k's jamming is non-zero.
2. The sender and receiver are synchronized so that bit level jamming is feasible.
3. The detection of jamming at bit level is hard.

Property 1 is basic to our work; we will examine its validity in an experimental sense in Section 6. Properties 2 and 3 are made for reasons for simplicity; we will discuss how they are relaxed in Sections 6 and 8.

Bit-Level Jamming Model. We consider a general bit level jamming model, which is shown in Table 2. In the model, a bit sent by j and jammed by k is decoded as 0 or 1 at e as follows. p is the probability of corrupting 0 to 1 (in other words, that a 0 sent by j is decoded as 1 by e), and q is the probability of corrupting 1 to 0. Note that this model is coarse-grain in the sense that it does not explicitly depend upon the bit value chosen by j for purposes of the cooperative jamming. If k chooses its jamming value (either 0 or 1) randomly, then $p = \frac{1}{2}(p^1_{0|\langle 0} + p^1_{0|\langle 1})$ and $q = \frac{1}{2}(p^0_{1|\langle 0} + p^0_{1|\langle 1})$. We will examine the validity of this model in Section 6.

j's bit	prob. e decodes 0	prob. e decodes 1
0	$1-p$	p
1	q	$1-q$

Table 2: Bit level jamming model where $p > 0$ and $q > 0$

Receiver Model. We consider two receiver models:

- **full-duplex**, where k knows x completely.
- **half-duplex**, where k knows x partially.

In the **full-duplex** model, k is able to jam and receive simultaneously. Hence, assuming that the communication channel is additive, k knows x completely because k fully knows its transmit and receive values. In this particular case, designing ϕ is easy. We discuss this model largely for pedagogical reasons.

The **half-duplex** model captures the case of a half-duplex transceiver where transmission can not happen simultaneously with reception. In this case, k knows x only partially because it only receives information when in listening mode. The half-duplex model is of greater practical interest than the full duplex model; we accordingly focus more on it in the paper.

Threat Model. The adversary of main interest for our problem of exchanging messages privately without using pre-shared secrets is simply a passive eavesdropper that knows the encoding and decoding functions. The eavesdropper may be assumed to have unlimited computational power. It may operate on devices of type other that of the wireless nodes j and k. Or, it may operate by compromising other wireless nodes; in this case, it may learn the node state of the compromised node.

Subsequently in the paper, we will extend the threat model to analyze how secure coding performs if the adversary has more knowledge about the jamming positions and values.

3. PROPERTIES OF SECURE CODING

To achieve perfect secrecy, any coding scheme must be able to tolerate the corruption of any location in x. Otherwise, if some positions in x are uncorruptible, the content of these locations will reveal some information about s to e. More precisely, we have:

PROPOSITION 1. *To achieve perfect secrecy, it must be that* $(\forall x : (\forall i : 1 \leq i \leq n : \ (\exists z : x_i \neq z_i)))$.

Next, we observe:

THEOREM 3.1. *For the* ***full-duplex*** *model, the maximal coding rate,* $\frac{m}{n} = 100\%$*, is achievable if* $\frac{1}{2} \leq p = q \leq 1$.

Let the encoding function be $f(x) = x$, and let k jam every bit in x with probability p' such that $p' \times p = \frac{1}{2}$, i.e., $p' = \frac{1}{2p}$. Now every bit that e receives could come from either 0 or 1 with equal probability, so e can correctly guess s with a probability of $\frac{1}{2^m}$. In this scheme, the coding rate is 100%.

THEOREM 3.2. *For the* ***half-duplex*** *model, the optimal coding rate in any scheme that achieves perfect secrecy is* 50% .

PROOF. We prove the theorem by contradiction. Suppose there exists an h ($h \geq 1$) such that $m = h$ and $n < 2h$ is sufficient to achieve perfect secrecy. Given an arbitrary s of length h, let A be the set of unjammed positions in the x of length n corresponding to s, and B be the set of jammed positions in x with respect to a jamming sequence γ. (Set A serves for recovery of s by k while B serves for confusion of e.) In order to recover s at k, it must be the case that $|A| \geq h$ and $|B| \leq h-1$.

Let γ^* be a jamming sequence whose corresponding set of unjammed positions, A^*, is of minimum size. (We refer to the corresponding set of unjammed positions of γ^* as B^*.) >From Proposition 1, we know that any position in x can be jammed, so we consider the effect on A^* if any one position A_i^* ($1 \leq i \leq |A^*|$) in x is jammed but the others in A^* are not. In order to recover s with this different jamming specification, since $|A^*|$ is minimum, it must be that some nonempty subset of positions in B^*, which we call C^i, are not jammed and become part of A^* in lieu of A_i^*. That is, to recover s, A_i^* and C^i are exchangeable regardless of the values taken on the remaining $|A^*| - 1$ positions. In other words, the value at the positions in C^i can be mapped to the value at A_i^* in terms of recovery (albeit the recovery of the value at position A_i^* may also be influenced by the values at some other positions in A^*).

This means that any one instance of values at the positions in A^* can be mapped from one instance of values taken from the positions in B^*. However, B^* can generate at most 2^{h-1} instances of values, while A^* requires at least 2^h instances of values. A contradiction. Therefore, $n \geq 2h$, i.e., $n \geq 2m$, and hence the optimal coding rate in the **half-duplex** model is 0.5. □

In Section 5, we will see a code example in the half-duplex receiver model that achieves the optimal coding rate for a particular jamming channel model.

4. PERFORMANCE OF EXISTING CODES UNDER JAMMING

In this section, we review how well existing coding techniques —specifically fountain codes (such as LT codes [13] and Raptor codes [19]) and secret sharing [18]— work for the secure problem. By analyzing their performance, we motivate the need to design a new class of codes, which we call dialog codes.

The secure coding problem can be reduced to the following well known secret sharing problem. Given m source symbols $s = \{s_1, \cdots, s_m\}$, design an encoder C that codes s into an n ($n \geq m$) symbol datum, such that k is able to recover s given any m from the n symbols. Any solution to this problem suffices to solve the secure coding problem if k can successfully jam $n - m$ of the symbols. Note that in the reduced problem, a symbol could be a bit, a byte, or a packet.

The reduced problem is well known to be solvable via fountain codes and secret sharing. In terms of analyzing their worst-case performance, we may assume that k is able to deterministically corrupt $n - m$ of the symbols. Since e can always randomly choose m symbols to decode with, its likelihood of guessing s is at least $\frac{1}{\binom{n}{m}}$. In order to guarantee perfect secrecy, we must have

$$\frac{1}{\binom{n}{m}} \leq \frac{1}{2^{t \times m}} \tag{1}$$

where t is the number of bits in each symbol. We show below that the coding rate is close to $\frac{1}{2^t}$.

We consider first how the coding rate, η ($\eta = \frac{m}{n}$ in this case), changes with t for a fixed value of m. Since $\binom{n}{m} = \binom{n+1}{m} \times \frac{n+1-m}{n+1}$, we have $\binom{n}{m} < \binom{n+1}{m}$. As t increases, from Inequation 1, n increases and consequently η decreases because of fixed m.

Now, let's consider how η changes with m for a fixed t. From Stirling's approximation [9], we have $n! = \sqrt{2\pi n}(\frac{n}{e})^n(1 + \Theta(\frac{1}{n}))$. Using this approximation, Inequation 1 reduces to

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

Although it follows that η increases when m increases, the upper bound of η also satisfies

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

where ϵ is a small value close to 0. Now, when t is relatively large, η has to be very small, and hence $log(\frac{1}{\eta}) \approx log(\frac{1}{\eta} - 1)$. While we have not derived a closed form for the upper bound of η, it follows from Inequation 3 that it must be close to $\frac{1}{2^t}$.

Thus, for example, for symbols with a size of 16 bits, secure coding with fountain codes or secret sharing leads to a coding rate of around $\frac{1}{2^{16}}$ to ensure perfect secrecy. For even smaller symbols, the upper bound of η is approximated in Figure 3.

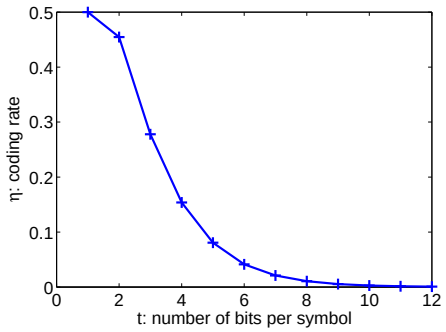

Figure 3: Upper bound of η with respect to t.

We see that the best coding rate we may achieve is 0.5 when $t = 1$. This means we should target a coding scheme on the bit level to minimize the coding overhead. However, the techniques we have considered (e.g., LT codes and Raptor codes) favor large symbols (usually a packet instead of a bit) in order be cost efficient, since each encoded symbol also brings overhead information such as the indexes of all the involved source symbols. They typically are used at a packet level (e.g., with potentially hundreds of bits). It follows that using these techniques would yield a very low coding rate if used to solve the secure coding problem. This motivates us to seek new coding approaches that are more efficient.

5. DIALOG CODES

5.1 Dialog Codes for the full-duplex model

In Theorem 3.1, we provided a simple solution for the special case of the full-duplex model where $0.5 \leq p = q \leq 1$. We present here a relatively simple coding function that suffices for the general full duplex model. The function encodes each source bit in s sequentially and independently at j to obtain x, as follows. First, the source bit is expanded to a t-bit word, a: $a = \{a_1, a_2, \cdots, a_t\}$, by adding a $t-1$ bit randomly chosen preamble to the source bit. Then, a is mapped to another t-bit word, b, using the function described in Table 3, which yields each b_i, $(1 \leq i \leq t)$ based on the value of both a_i and a_{i-1}. By way of definition, we let $a_0 = 0$ and assume this default value is well known.

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

Table 3: Mapping for second stage of the full-duplex encoder. Each bit is encoded using its own value and its 1-bit history value.

As an example, the 4-bit word $a = 0101$ is thus mapped to $b = 1000$. Figure 4 illustrates the encoder.

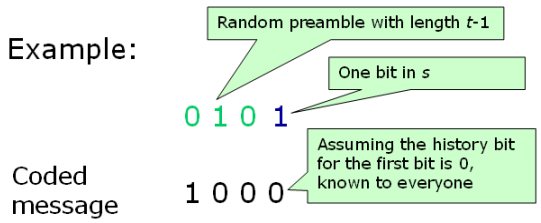

Figure 4: An example of the dialog code for the full duplex model

The jamming sequence used by k in the full duplex model consists of all bit positions in x. Since each source bit is encoded independently, we observe:

THEOREM 5.1. *When k jams every bit in b, the probability of correctly guessing a_t by e is upper bounded by $\frac{1}{2}(1 + (1 - 2w)^t)$, $w = min(p, 1 - p, q, 1 - q)$, which converges to $\frac{1}{2}$ as t increases.*

The proof of this observation is relegated to the Appendix. Since e can guess each source bit with probability up to 0.5, the probability of correctly guessing s therefore converges to $\frac{1}{2^m}$. Note that as t increases the coding rate for this scheme, $\eta = \frac{1}{t}$, decreases but on the other hand the secrecy level increases.

5.2 Dialog Codes for the half-duplex model

In the **half-duplex** model, k does not know the original value of jammed fields. It does however know the position of the jammed fields in the information it receives in the message y, and needs to recover the input s accordingly.

To motivate the design of the dialog code for the general half-duplex model, we start with a simple, efficient dialog code that suffices for the special case of the model where $p = q = 1$ (i.e., the flipping model where corruption is deterministic when jamming occurs) and then generalize that code.

5.2.1 A Solution for the Flipping Model

Let each bit in s be represented by two bits as follows,

$$x_{2i-1}\ x_{2i} = \begin{cases} 0\ 0 & if\ s_i = 0 \\ 1\ 1 & if\ s_i = 1 \end{cases} \quad (4)$$

k's strategy is to jam either position of each pair, and to recover the input simply by looking at the remaining bit of each pair. Since the corruption upon jamming is deterministic, what e sees would be always either 01 or 10, which may come from either 11 or 00 equally, so the probability for e to make a correct guess for each pair is $\frac{1}{2}$. Therefore, e's chance to correctly guess s is $\frac{1}{2^m}$, which means that perfect secrecy is achieved. Note also that according to Theorem 3.1, this scheme achieves the optimal coding rate.

By way of an example, this time letting $s = 1101$, s would be encoded as 11 11 00 11 by j. If k were to corrupt the first bit in each pair, then e would receive the corrupted value 01 01 10 01 at e and ?1 ?1 ?0 ?1 would be received at k.

Of course, this special case does not deal with corruption failures. If a corruption on any pair of bits fails, an occurrence of 00 or 11 helps e to discover these bits directly. Moreover, in general, both p and q may not be 1, and may change over time. Fortunately, an efficient generalization of this scheme works for the general half-duplex model.

5.2.2 A Solution for the General Model

The basic idea is the same as that of the full-duplex model: each bit in s is encoded sequentially and independently. But, instead of using a one-bit history in the second stage of the encoding (after expanding the source bit with a $t-1$ preamble), we now use a 2-bit history. Specifically, each t-bit word a is encoded to $b = \{b_1, b_2, \cdots, b_{2t}\}$ following the rule in Table 4. We let $a_{-1}a_0 = 00$.

$a_{i-2}\ a_{i-1}$	$a_i = 0$ $b_{2i-1}\ b_{2i}$	$a_i = 1$ $b_{2i-1}\ b_{2i}$
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 4: Encoding scheme for the half-duplex model

Continuing our previous example, an input word $a = 0101$ would be encoded in this case as 01 10 11 01. Figure 5 illustrates this encoder.

The jamming sequence used by k is again to randomly choose one of the bits in each pair to corrupt. The remaining bit is sufficient for k to recover s. The decoding procedure thus has $O(1)$ complexity per bit. Also, since j creates an output x of size n ($n = 2tm$), the coding rate is $\frac{m}{n} = \frac{1}{2t}$, and since a table lookup suffice, the coding procedure has $O(1)$ complexity per bit. Since each source bit is still independently encoded, we observe:

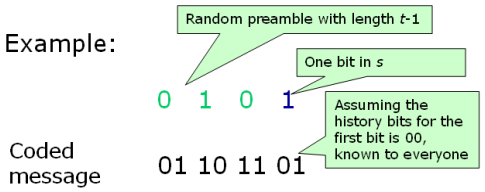

Figure 5: An example of the dialog code for the half duplex model

THEOREM 5.2. *When k jams one bit in each pair in b, the probability that e correctly guesses a_t is upper bounded by $\frac{1}{2}(1 + (1-w)^{\frac{t+1}{2}})$, $w = min(p, q)$, which converges to $\frac{1}{2}$ as t increases.*

The proof of this observation is also relegated to the Appendix. Although Equation 10 in the proof does not yield a closed form for a tight upper bound on the convergence speed, in practice, the convergence speed is much faster than $\frac{1}{2}(1 + (1-w)^{\frac{t+1}{2}})$.

To gain intuition of how to choose the preamble length for different values of p and q, we plot in Figure 6 the least preamble length that ensures the probability of guessing the source bit is in the interval [0.49, 0.51] versus $w = min(p, q)$. We see for instance that a 6-bit preamble is sufficient to confuse e when $w = 0.5$. Also, that if w is large, then the length of the preamble becomes relatively short. (And indeed, when both p and q are 1, the preamble length is zero and the scheme achieves the optimal coding rate of 0.5.)

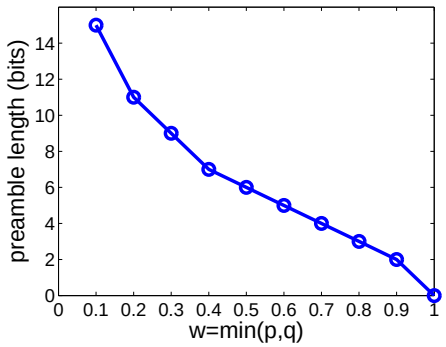

Figure 6: Length of preamble required to confuse e ($|l_t - 0.5| < 0.01$)

In practice, since p and q may change over time, it is best to choose the preamble length conservatively. At

the same time, note that greater variability in p and q is helpful in the sense that it makes it harder for e to guess correctly based on its prior knowledge of p and q.

Remark. In situations where perfect secrecy is not strictly required, the preamble can be skipped entirely while still obtaining a high level of secrecy. For example, if $p = q = 0.5$ and we use this dialog code with $t = 1$, the probability of e correctly guessing a 29-byte message (which is a typical packet size in TinyOS) is less than $\frac{1}{2^{96}}$. Such a security strength may well be sufficient for a number of embedded wireless sensor network applications. To appreciate this, recall that the security level achieved with even a 1024-bit key using asymmetric RSA is approximately $\frac{1}{2^{80}}$ [1]. *End of remark.*

6. EXPERIMENTAL VALIDATION

The goal of the experimental study in this section is to validate our jamming model and the assumption of predictable jamming, as well as to explore the feasibility of implementing the dialog codes on conventional wireless sensor network platforms such as the *CC1000* motes and *CC2420* motes [14].

Our experimental study has the following general characteristics. We have used TinyOS [2] as the software platform, and channel 2 on $CC1000$ motes and channel 11 on $CC2420$ motes in all our experiments. We have disabled the default CSMA protocol in TinyOS to allow for intentional interference. We let one node trigger the communication between j and k, i.e., both j and k start sending a message immediately after they receive this control message. To finesse this, we modified the TinyOS library so that the *send* command and *receive* event are triggered immediately without posting too many tasks.

Because of limited space, we present only some representative experiments below. A complete experimental study may be found in an accompanying technical report [3].

6.1 Experimental Setup

We chose four topologies *(a)*, *(b)*, *(c)* and *(d)*, as shown in Figure 7, for our experiments. e and j are physically abutting each other in topologies *(a)* and *(b)*: e is on top of j in one and side-by-side in the other. The reason for this choice is to test how jamming performs in this worst-case from the perspective of locating e (since e should intuitively be able to overhear j's messages best at these locations). In *(c)*, we moved e from j towards k to see how distance impacts corruption. In *d*, we moved k towards e to see the effects of jamming on not only "inner-band"links (which have high packet delivery ratio) but also for "middle-band" links (which have modest packet delivery ratio).

We then carried out the following experiments:

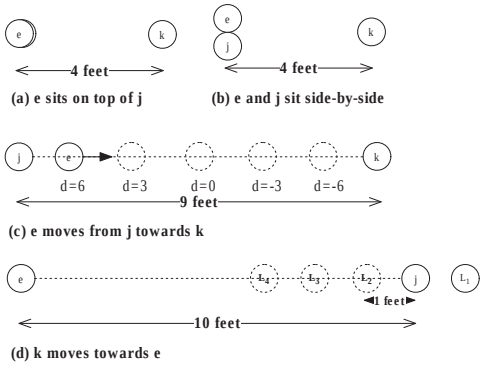

Figure 7: Experimental topologies, $d = dist(e, j) - dist(e, k)$

1. **Fixed location, varying jamming power:** In this experiment, we used topology *a*. j sent a message containing all zeros at power level 3 (-24 dBm) immediately after receiving e's control message; meanwhile, k sent a message containing all ones at various power levels, ranging from 3 to 31 (0 dBm).

2. **Fixed power, varying e's location:** In this experiment, we used topology *c*. We collected data at e as we moved it from j towards k, where $d = \{6, 3, 0, -3, -6\}$ feet. j sent all zeros and k jammed with all ones, both at power level 3.

3. **Varying the bit value:** In this experiment, we used topology *a* and *b*. j used power level 3, and k used power level 31. We recorded data for different bit sequence sent by j and j as follows, (I) j: 0's; k: 1's; (II) j: 1's; k: 0's; (III) j: 0's; k: 0's; (IV) j: 1's; k: 1's.

4. **Middle-band links:** In contrast to the previous three experiments, which were conducted on inner-band links where the PRR between j and e is more than 99% in the absence of k, here we located j and e such that the average PRR was below 80% in the absence of k. We used topology *d*, and located k at positions L_1, L_2, L_3, and L_4, to measure the loss and corruption difference when k used power level 3 for its jamming.

In each experiment, e triggered communication once every 300 milliseconds, for a total of 30 minutes. The packet size was 64 bytes in all experiments.

6.2 Validating the Jamming Model

According to the SINR (*Signal to Interference plus Noise Ratio*) model (aka the physical model [8]), whether or not a transmission is successful depends on the received signal strength, the interference caused by simul-

taneous transmissions from other nodes, and the ambient noise level. That is, a transmission is successful iff $\frac{P_e}{P_n+I_e} \geq \beta$ where P_e is the power of a signal sent by j when received at e, P_n is the noise power level, I_e is the interference power generated by other concurrent transmissions, and β is the minimum signal-to-interference-ratio that is required for a message to be delivered at a receiver.

Since received power is typically modeled as decaying exponentially with the distance of $dist(j,e)$, i.e., $P_e = \frac{P_j}{dist(j,e)^\alpha}$ and the minimum interference power in the presence of jamming is the received power at e from k, successful jamming needs:

$$\frac{P_k}{dist(k,e)^\alpha} > \frac{P_j}{\beta \times dist(j,e)^\alpha} - P_n \quad (5)$$

A system designer can thus estimate the unknown parameters in Equation 5 to provide a procedure for meeting the goals of jamming by k. Note that even if the eavesdropper e is a "hidden terminal" with respect to k, jamming can be still successful as long as k's messages can disrupt j's messages from being correctly received at e. To provide added confidence beyond analytically exploring the impact of each parameter on the outcome of jamming, we now profile the jamming outcomes experimentally.

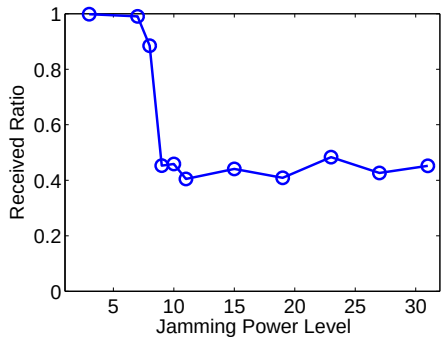

Figure 8: Bit level corruption when j sends 0's and k sends 1's, $p^0_{0\lhd 1}$, at different jamming power levels

Figure 8 shows the bit level corruption statistics collected in Experiment 1. $p^0_{0\lhd 1}$ remains relatively stable in the presence of jamming. If corruption is low (e.g., jamming power level is less than 9), $p^0_{0\lhd 1}$ is certainly dominated by uncorrupted message 0's. However, when more messages are corrupted, $p^0_{0\lhd 1}$ increases and stays within a certain range, [0.4 .. 0.6], in this data set. This shows that it is feasible to achieve a non-trivial probability of corrupting any bit via jamming.

Table 5 shows the bit level corruption statistics collected in Experiment 2. Again, we see that the probability of corrupting a bit is non-trivial. Specifically, most of $p^0_{0\lhd 1}$ are around 50%, somewhat independent of the location of e.

	$p^0_{0\lhd 1}$
$d = 6$	55.96%
$d = 3$	51.86%
$d = 0$	50.85%
$d = -3$	37.69%
$d = -6$	46.56%

Table 5: Bit level corruption when e moves from j to k, $d = dist(e,j) - dist(e,k)$

	side-by-side	one on the other
$p^0_{0\lhd 1}$	45.05%	46.95%
$p^0_{1\lhd 0}$	42.81%	40.37%
$p^0_{0\lhd 0}$	15.63%	15.58%
$p^0_{1\lhd 1}$	52.55%	53.68%

Table 6: Bit level corruption where j and e vary bit values, showing non-trivial probability of bit corruption

Table 6 shows the bit level corruption statistics in Experiment 3. Again, $p^0_{0\lhd 0}$ is non-trivial regardless of what bit values j and k sent.

Experiment 4 showed that when the link between j and e is in the middle-band, i.e., the packet delivery rate is modest, then even jamming at a moderate power level is sufficient to confuse e. As the jamming power level increases, say because k moves closer to e, j's messages are corrupted more and after some point k's transmission values start dominating in the messages received at e.

In addition to these experiments on *CC2420* motes, we also experimented on MICA2 motes (with *CC1000* radio). Since the results were consistent with those observed on *CC2420* motes, we present only one set of results here. In these experiments, both j and k are located around 4 feet away from e, and both use power level 3. The bit corruption results are shown in Table 7. Again, we see that the probability of corrupting a bit by jamming is fairly high, at least 37% in this data set.

Not only is the probability of corrupting a bit non-trivial, we find that this probability exhibits significant temporal variation, as shown in Figure 9. Most cases have around 15% (absolute) variation, while some change dramatically over time. For example, $p^0_{0\lhd 0}$) (top) is around 20% in the first 50-second period, but it jumps to 70% and stays around there after that. $p^0_{1\lhd 1}$) (top)

$p^0_{0\not\lhd 1}$	37.74%
$p^0_{1\not\lhd 0}$	65.68%
$p^0_{0\not\lhd 0}$	65.27%
$p^0_{1\not\lhd 1}$	37.77%

Table 7: Bit level corruption using *CC1000* motes

has even larger fluctuation. It varies between 0% and 90%. As noted previously, increased variation makes predicting $p^0_{1\not\lhd 1}$ even harder, which helps the secrecy level between j and k.

We relegate the details of the possibility of detecting bit-level jamming via signal strength and network recovery to the jamming work [16]. In summary, we observed that it is hard to discriminate corrupted packets (introduced by jamming) from uncorrupted packets using the signal strength only. Bit-level detection would be even harder. We also observed that not only can a single receiver not recover source packets correctly in the presence of jamming, but also a network (a set of receivers) can not recover the source packets based on simple counting logic, that said, whether it is possible to do network based recovery using more complicated methods is still an open question.

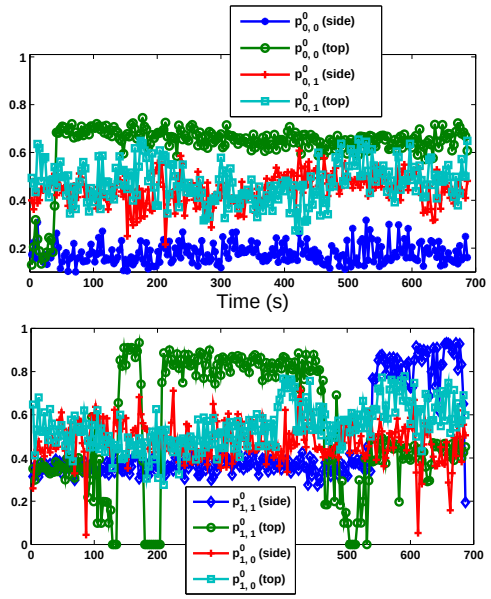

Figure 9: Bit level corruption changes over time ($p^w_{u,v}$ abbreviates $p^w_{u\not\lhd v}$)

In summary, we conclude that there it is feasible to achieve a non-trivial probability of corrupting a bit via jamming in these platforms. Moreover, the corruption probability also changes over time, sometimes even dramatically. When jamming is effective, e does not necessarily benefit from sitting closer to the sender than k, nor does e benefit much when k lowers its jamming power.

6.3 Implementing Dialog Codes

Currently, we are not in a position to implement the dialog codes at the level of bits, say by modifying the physical layer in an appropriate mote/software/cognitive radio platform. We have however implemented dialog coding at the level of bytes on the *CC1000* motes; by byte-level, we mean that we use a byte to represent a single bit, by letting the XOR value of all bits in that byte denote the intended bit. Likewise, we have implemented dialog coding at the level of whole packets on the *CC2420* motes using TinyOS. These implementations enable a preliminary experimental evaluation of the validity of dialog codes and an exploration of synchronization requirements between j and k.

We describe here our byte-level implementation (the packet-level is similar).

1. j encodes s using the procedure in Section 5.2.2. Each resulting bit is expanded randomly into a byte whose XOR matches the bit.

2. During transmission, j precedes each byte with two "sync bytes" so that k can synchronize to jam that data byte.

3. Upon receiving the first sync bytes, k randomly injects a"jamming byte" in time to corrupt one of the next two data bytes, and records the position of the jammed data byte as well collects the other data byte.

The following trace shows that while k was able to decode a single message repeatedly sent by j, evader e was not able to decode the message even once in this trace. (The size $t = 4$ was chosen for deciding the preamble length in the encoding for this trace of length 100 messages; also, e simply discarded the sync bytes before it tried to decode the message.)

Repeated input at j:

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
...

7. RELATED WORK

Information theory. As mentioned in the introduction, there has been substantial interest in recent years in physical layer security in wireless communications.

Building on Wyner's seminal result about the possibility of secure communication when the eavesdropper's channel is degraded with respect to the legitimate receiver's channel, diverse analysis has been performed for the capacity of confidential unicast: some models exploit feedback on receiver/channel state [11, 20], some exploit the ability of the receiver to selectively jam the transmitter at secretly chosen times, some exploit multipath [17], and others exploit power level selection. By the same token, diverse models for confidential multicast and unicast over untrusted relays [10] have also been considered. The focus thus far largely has been on theory.

Coding theory and cryptology. Fountain codes (aka rateless erasure codes), including LT codes [13] and Raptor codes [19], 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. As discussed earlier, they may be used for secure coding albeit with low coding rate, since they solve the secret sharing problem [4, 18].

Unlike the secret sharing problem, where all pieces of information made available to the receiver are accurate, the verifiable secret sharing problem [6] lets each player verify the shared secret even though a certain amount of its information may be inaccurate. A simple version of this problem is as follows: A player receives n pieces of information about the dealer's secret, of which k pieces are false. A verifiable scheme should allow the player to reconstruct the dealer's secret given the n pieces information. This is in contrast to the secure coding problem, where e knows that at most $n-m$ bits are corrupted, but does not know which bits are corrupted and should not be able to reconstruct the n bit information; on the other hand, k which knows which bits are corrupted and should be able to successfully reconstruct the information.

There is a significant body of work in the coding theory for the specific wiretap channel. Ozarow and Wyner [15] showed that the maximum number of symbols that the source can communicate to the receiver securely in the information theoretic sense is equal to $n - \mu$ under Ozarow-Wyner wiretap channel of type II. Wei [22] generalized Hamming weights for linear codes and characterized the code performance on the wire-tap channel of type II. Thangaraj et al [21] proposed specific codes based on LDPC codes for binary erasure channel and binary symmetric channel. The authors in [5] introduced a paring protocol over anonymous channel to establish shared secrets.

Lastly, it is worthwhile to recall that the jamming approach is similar to the one-time pad (OTP) in the sense that overheard messages are randomly produced by source messages and jamming messages. Of course, unlike OTP, it does not require secure generation and exchange of the one-time pad material.

8. DISCUSSION AND CONCLUSION

This paper has taken a first step in the development of codes for confidential communications at the physical layer. We have defined the secure coding problem, wherein perfect secrecy is achieved in communication without necessarily using any shared secrets. We have designed dialog codes, which solve this problem with O(1) computation complexity in both the encoding and the decoding process. The coding rate of these codes is optimal for specific jamming models. Some code work correctly in general jamming models and are notably tolerant to channel variability over time; when used with optimal coding rate in the general half-duplex model (i.e., with zero-length preambles) they achieve high — albeit not perfect— security.

We have demonstrated the feasibility of dialog codes at both the byte level and the packet level on wireless sensor network platforms, respectively the *CC1000* motes and *CC2420* motes, and tested the security of our implementations in small scale experiments.

Our exposition has assumed that detection of jamming at a bit-level is difficult. Note that in the secure coding problem, unlike external or non-cooperative jamming, the receiver can cooperate with the sender in choosing the respective transmission parameters in order to make this detection hard. Nonetheless, while this assumption is arguably applicable in extant mote platforms, one can envision sophisticated cognitive/software radio platforms where this assumption will not hold. In these platforms, dialog codes would not suffice for perfect security. This leads us to explore whether how to achieve perfect security in confidential communications without this assumption.

Consider a symmetric formulation of secure coding, where both j and k wish to simultaneously exchange their respective inputs, s_j and s_k. Both j and k concurrently send to each other their encoded inputs according to a jamming protocol. To motivate how they jam each other, we provide a simple example. Given 3 bit slots, both j and k choose a slot for sending one bit each. The sends collide $\frac{1}{3}$ of the time, while $\frac{2}{3}$ of the time, both bits get through. Although e can determine whether jamming occurred in a slot, it does not know who communicated in which position $\frac{2}{3}$ of the time. Although this example does not illustrate a bounded length exchange protocol, we note in passing that the symmetric secure coding problem has a bounded length solution.

There are a number of avenues of further research. The implementation of bit-level physical layer dialog

codes is one. The exploration of the relative merits of alternative codes which do not code at the level of individual bits is another. The exploration of how the security strength of dialog codes degrades as the eavesdropper gains in its ability to detect corruption is also of interest. Finally, our exposition has assumed that synchronization at the bit level is feasible, although our experiments have only dealt with synchronization at the level of bytes or packets. On the one hand, bit level synchronization at the physical layer needs further research. On the other hand, when bit-level synchronization occasionally fails (or eavesdroppers become malicious jammers themselves) insertions (as opposed to corruptions) occur, which would need to be handled in the implementation just as losses would need to be handled, possibly using mechanisms such as checksums.

9. REFERENCES

[1] http://en.wikipedia.org/wiki/key_size.

[2] http://www.tinyos.net.

[3] A. Arora and L. Sang. Dialog codes for secure wireless communications. *Technical Report, Ohio State University, OSU-CISRC-5/08-TR23*, 2008.

[4] G. Blakley. Safeguarding cryptographic keys. *Proc. AFIPS 1979 NCC*, 48:pp. 313–317, June 1979.

[5] C. Castelluccia and P. Mutaf. Shake them up!: a movement-based pairing protocol for cpu-constrained devices. *Mobisys*, 2005.

[6] 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.

[7] I. Csiszar and J. Korner. Broadcast channels with confidential messages. *IEEE Trans. Information Theory*, 24(3):339–348, May 1978.

[8] P. Gupta and P. Kumar. The capacity of wireless networks. *IEEE Transaction on information theory*, 46(2), Mar 2000.

[9] T. H.Cormen, C. E.Leiserson, R. L.Rivest, and C. Stein. Introduction to algorithms. *The MIT Press*, 2001.

[10] X. He and A. Yener. The role of an untrusted relay in secret communication. *in Proceedings of the IEEE International Symposium on Information Theory (ISIT'08)*, 2008.

[11] L. Lai, H. E. Gamal, and H. V. Poor. The wiretap channel with feedback: Encryption over the channel. *IEEE Transactions on Information Theory*, 54(11):5059 – 5067, Nov 2008.

[12] S. K. Leung-Yan-Cheong and M. E. Hellman. The gaussian wire-tap channel. *IEEE Trans. Information Theory*, 24(4):451–456, July 1978.

[13] M. Luby. Lt codes. *Foundations of Computer Science, 2002.*, pages 271 – 280, 2002.

[14] MoteIV. http://www.moteiv.com/.

[15] L. H. Ozarow and A. D. Wyner. The wire-tap channel ii. *Bell Syst. Tech. Journal*, 63:2135–2157, 1984.

[16] L. Sang and A. Arora. Capabilities of low-power wireless jammers. *Infocom Miniconference*, 2009.

[17] A. Sayeed and A. Perrig. Secure wireless communications: Secret keys through multipath. *ICASSP*, 2008.

[18] A. Shamir. How to share a secret. *Communications of the ACM*, 22(11):612 – 613, 1979.

[19] A. Shokrollahi. Raptor codes. *IEEE Transactions on Information Theory*, 52(6):2551 – 2567, 2006.

[20] E. Tekin and A. Yener. The general gaussian multiple access and two-way wire-tap channels: Achievable rates and cooperative jamming. *IEEE Transactions on Information Theory - Special Issue on Information Theoretic Security*, 54(6):2735–2751, 2008.

[21] A. Thangaraj, S. Dihidar, A. R. Calderbank, S. W. McLaughlin, and J.-M. Merolla. Applications of ldpc codes to the wiretap channel. *IEEE Transactions on Information Theory*, 53(8):2933–2945, 2007.

[22] V. K. Wei. Generalized hamming weights for linear codes. *IEEE Transactions on Information Theory*, 37(5):1412–1418, 1991.

[23] A. D. Wyner. The wire-tap channel. *in Bell Syst. Tech. J., 54(8)*, pages 1355–1387, 1975.

10. APPENDIX

Proof of Theorem 5.1:

THEOREM 5.1. *When k jams every bit in b, the probability of correctly guessing a_t by e is upper bounded by $\frac{1}{2}(1+(1-2w)^t)$, $w = min(p, 1-p, q, 1-q)$, which converges to $\frac{1}{2}$ as t increases.*

PROOF. We begin with the case of $p = q$ and $p \leq 0.5$. Let $a^* = \{a_1^*, a_2^*, \cdots, a_t^*\}$ denote the message inferred by e when j sends a and $Pr(a_{i-1}^* = 0) = l_{i-1}$. Suppose j sends $a_i = 0$, the probability for e to correctly guess a_i is as follows,

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

$$l_i = Pr(a_i^* = 0) = (1-p) \times l_{i-1} + p \times (1-l_{i-1}) \quad (6)$$

(The calculation for $a_i = 1$ is similar.)

Letting $u_i = l_i - \frac{1}{2}$, Equation 6 now becomes $u_i = u_{i-1}(1-2p) = u_0(1-2p)^i$. Since the history bit a_0 is

well known known, $l_0 = 1$, and hence $u_i = \frac{1}{2}(1-2p)^i$, i.e., $l_i = \frac{1}{2}(1+(1-2p)^i)$.

Note that p has the same impact on l_i as $1-p$ if e flips every bit it receives. So does q. When $p \neq q$, we let $w = min(p, 1-p, q, 1-q)$, and substituting p with w in Equation 6, we see that the convergence property still holds. Therefore, the probability that e correctly guesses a_t is upper bounded by $\frac{1}{2}(1+(1-2w)^t)$, $w = min(p, 1-p, q, 1-q)$, which converges to 0.5 if $0 < p, q < 1$. □

Proof of Theorem 5.2:

THEOREM 5.2. *When k jams every bit in b, the probability of correctly guessing a_t by e is upper bounded by $\frac{1}{2}(1+(1-2w)^t)$, $w = min(p, 1-p, q, 1-q)$, which converges to $\frac{1}{2}$ as t increases.*

PROOF. Without loss of generality, let $a_{i-2} = 0$, $a_{i-1} = 0$ and $a_i = 0$. In this case, a_i is encoded as 01 according to the encoding scheme in Table 4. Let $c = (c_1, c_2, \cdots, c_{2t})$ be the corrupted copy of b received at e, and $a^* = (a_1^*, a_2^*, \cdots, a_t^*)$ be the message inferred by e about a upon receiving c. Since k may equally choose either bit to jam in each pair, 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 two history bits at e is

$$\begin{aligned} &Pr(a_{i-2}^* = 0, a_{i-1}^* = 0) = d_1, \\ &Pr(a_{i-2}^* = 0, a_{i-1}^* = 1) = d_2, \\ &Pr(a_{i-2}^* = 1, a_{i-1}^* = 0) = d_3, \\ &Pr(a_{i-2}^* = 1, a_{i-1}^* = 1) = d_4 \end{aligned}$$

where $d_1 + d_2 + d_3 + d_4 = 1$ and $0 \leq d_1, d_2, d_3, d_4 \leq 1$.

Consider the case where e infers $a_{i-2}^* = 0$ and $a_{i-1}^* = 0$. When e receives 01, e guesses correctly with probability $d_1 \times (1 - \frac{p+q}{2})$. When e receives 11 or 00, e knows that there must be a corruption since no such a mapping exists according to the mapping table given the history 00, so e has to guess this pair. The probability that correctly guesses $a_i = 0$ is $d_1 \times (p+q)/4$ in this case. Therefore, the probability that e knows $a_i = 0$ is $d_1 \times (1 - \frac{p+q}{4})$. Similarly, 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 correctly guessing a_i 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 $d_1 = d_2 = d_3 = d_4 = 1/4$; in this case, our proof is complete. If d_1, d_2, d_3, and d_4 are not equal, consider the marginal probability of the 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:

$$\begin{aligned} &Pr(a_{i-1}^* = 0, a_i^* = 0) = (d_1 + d_3) \times l_1, \\ &Pr(a_{i-1}^* = 0, a_i^* = 1) = (d_1 + d_3) \times (1 - l_1), \\ &Pr(a_{i-1}^* = 1, a_i^* = 0) = (d_2 + d_4) \times l_1, \\ &Pr(a_{i-1}^* = 1, a_i^* = 1) = (d_2 + d_4) \times (1 - l_1) \end{aligned}$$

Letting $l_0 = d_1 + d_3$, the above equations become:

$$\begin{aligned} &Pr(a_{i-1}^* = 0, a_i^* = 0) = l_0 \times l_1, \\ &Pr(a_{i-1}^* = 0, a_i^* = 1) = l_0 \times (1 - l_1), \\ &Pr(a_{i-1}^* = 1, a_i^* = 0) = (1 - l_0) \times l_1, \\ &Pr(a_{i-1}^* = 1, a_i^* = 1) = (1 - l_0) \times (1 - l_1) \end{aligned} \tag{7}$$

Let us first consider the case where $a_{i+u} = 0$ for $u \geq 0$. Then the marginal probability can be generalized as:

$$\begin{aligned} &Pr(a_{i+u}^* = 0, a_{i+u+1}^* = 0) = l_{u+1} \times l_{u+2}, \\ &Pr(a_{i+u}^* = 0, a_{i+u+1}^* = 1) = l_{u+1} \times (1 - l_{u+2}), \\ &Pr(a_{i+u}^* = 1, a_{i+u+1}^* = 0) = (1 - l_{u+1}) \times l_{u+2}, \\ &Pr(a_{i+u}^* = 1, a_{i+u+1}^* = 1) = (1 - l_{u+1}) \times (1 - l_{u+2}) \end{aligned} \tag{8}$$

where

$$\begin{aligned} l_{u+2} &= l_u l_{u+1}(1 - \frac{p+q}{4}) + l_j(1 - l_{u+1})(\frac{1}{2} + \frac{p-q}{4}) \\ &+ (1 - l_u) l_{u+1} \frac{p+q}{4} + (1 - l_u)(1 - l_{u+1})(\frac{1}{2} + \frac{q-p}{4}) \\ &= (l_{u+1}(1-p) + \frac{p-q}{2})(l_u - 1/2) + 1/2. \end{aligned} \tag{9}$$

Letting $v_u = l_u - 1/2$ where $-1/2 \leq v_u \leq 1/2$, the above equation becomes:

$$v_{u+2} = v_u \times (v_{u+1}(1-p) + \frac{1-q}{2}) \tag{10}$$

It is easy to check that v_u converges to 0. Let $w = min(p, q)$, if $v_0 > 0$, we have $v_{u+2} \leq v_u(v_{u+1} + \frac{1}{2})(1 - w) \leq v_u(1-w) \leq max((1-w)^{(u+2)/2} \times v_0, (1-w)^{(u+1)/2} \times v_1)$. When $v_0 < 0$, v_u also converges to 0 by a similar argument. The same conclusion can be likewise drawn for arbitrary a_i, with slight changes in Equation 8.

Therefore, the probability for e to correctly guess s_i, or a_t, is upper bounded by $\frac{1}{2}(1 + (1-w)^{\frac{t+1}{2}})$, which converges to 1/2. □