

# Integrity Codes: Message Integrity Protection and Authentication Over Insecure Channels

Srdjan Čapkun<sup>†\*</sup>    Mario Čagalj<sup>§\*</sup>    Ramkumar Rengaswamy<sup>‡</sup>  
 Ilias Tsigkogiannis<sup>‡</sup>    Jean-Pierre Hubaux<sup>¶</sup>    Mani Srivastava<sup>‡</sup>

<sup>†</sup>Department of Computer Science, ETH Zurich, 8092 Zurich, Switzerland

<sup>§</sup>FESB, University of Split, 21000 Split, Croatia

<sup>¶</sup>LCA, Swiss Federal Institute of Technology Lausanne (EPFL), 1015 Lausanne, Switzerland

<sup>‡</sup>Networked and Embedded Systems Laboratory, University of California Los Angeles, USA

capkuns@inf.ethz.ch    mario.cagalj@fesb.hr    ram@ucla.edu    ilias@ee.ucla.edu  
 jean-pierre.hubaux@epfl.ch    mbs@ucla.edu

**Abstract**—Inspired by unidirectional error detecting codes that are used in situations where only one kind of bit error is possible (e.g., it is possible to change a bit “0” into a bit “1”, but not the contrary), we propose integrity codes (*I*-codes) for radio communication channels, which enable integrity protection of messages exchanged between entities that do not hold any mutual authentication material (i.e. public keys or shared secret keys).

The construction of *I*-codes enables a sender to encode any message such that if its integrity is violated in transmission over a radio channel, the receiver is able to detect it. In order to achieve this, we rely on the physical properties of the radio channel and on unidirectional error detecting codes. We analyze in detail the use of *I*-codes and we present their implementation on a wireless platform as a “proof of concept”. We further introduce a novel concept called “authentication through presence”, whose broad applications include broadcast authentication, key establishment and navigation signal protection. We perform a detailed analysis of the security of our coding scheme and we show that it is secure within a realistic attacker model.

**Index Terms**—Broadcast Authentication, Integrity, Key agreement protocols, Man In the Middle Attacks (MITM), Wireless networks

## I. INTRODUCTION

Conventional security goals like message confidentiality, integrity, and authentication are traditionally achieved through the use of certified public-keys or shared secret keys, and by the application of appropriate cryptographic primitives (i.e., encryption schemes, signatures, message authentication codes, etc.).

Attacks on the integrity of messages exchanged over a wireless communication channel include typically message modification by bit flipping (modification of one of several bits of the original message by the attacker) and message overshadowing (an original signal appears as noise in a much stronger attacker’s signal, which is then accepted as valid by the receiver). Examples of message overshadowing attacks can be found in the context of IEEE 802.11 access point SSID hijacking [30]. Bit flipping and message overshadowing attacks cannot be detected by relying solely on (non-keyed) error correcting codes. A bit flipping attacker can change the transmitted message and the code appended to the message such that the modification is not detected at the receiver. Likewise, message overshadowing disables the reception of the original message (and its correcting code) at the receiver; the receiver receives the attacker’s message and its code. These

attacks demonstrate that the detection of the modifications of the messages transmitted over a wireless communication channel will not be possible using only error detecting/correcting codes, unless the communication parties share a secret key based on which message coding will be keyed and thus will provide message authentication and integrity protection (e.g., using Message Authentication Codes (MAC) or digital signatures).

In this paper, we propose *I*-codes, a new security primitive that enables authentication and integrity protection of messages exchanged over a radio communication channel between entities that do not hold any shared secrets or mutual authentication material (i.e. public keys or shared secret keys). The construction of *I*-codes enables a sender to encode any message, such that if its integrity is violated in transmission, the receiver is able to detect it. *I*-codes consist of three components: unidirectional message coding, on-off keying communication, and receiver’s awareness of presence in the sender’s transmission range. *Unidirectional Error-Detecting* codes [7], [9], [18], [8] are used in situations where it is possible to change, for example, a bit “0” into a bit “1” but the contrary is not possible (except with a negligible probability). Unidirectional error-detecting code is able to detect any number of unidirectional errors in a given codeword; in other words, for a given error-detection code, no unidirectional error can transform a (valid) codeword into another (valid) codeword. On-off keying is a modulation by which the bit “1” is transmitted on the channel as the *presence* of a signal and the bit “0” is transmitted as the *absence* of a signal. Signal anti-blocking means that the energy of the signal (bit “1”) cannot be annihilated by an adversary (we show several ways how to ensure this). On-off keying therefore creates a modulation scheme supporting the usage of uni-directional error-correcting codes.

Composed of these three components, *I*-codes enable (broadcast) message *authentication through presence awareness*. In other words, in order to guarantee that the received message is authentic and that its integrity was not violated, the receiver needs to be aware of its presence in the transmission range of the sender and does not need to share any secrets with the sender or hold any certificates related to sender’s public key. Ensuring integrity protection over insecure radio channels is important for preventing “man-in-the-middle” attacks, which could otherwise be perpetrated on the radio channel. By taking advantage of the characteristics of the radio channel, the *I*-codes help to completely prevent this attack. We show the application of authentication through presence through three examples: (1)

\*Equally contributing authors.

IEEE 802.11 access point authentication, (2) key establishment over insecure radio channels and (3) authentication of navigation signals. Through these applications we show that *I*-codes enable not only message origin authentication, but also effectively detect attacks pulse-delays and replay attacks on localization and synchronization [34].

We perform a detailed analysis of the security of *I*-codes on a radio channel and we show that they are secure assuming a realistic attacker model. This analysis takes into account the characteristics of the radio channel such as phase shifts, noise, and the attacker's ability to detect, jam and alter the messages on the channel.

To validate our concept, we implement and test *I*-codes on a number of wireless platforms. Our implementations demonstrate that *I*-codes can be efficiently implemented using existing radio and processing hardware.

The paper is organized as follows. In Section II, we state our problem and we describe our system and the attacker model. In Section III, we formally introduce *I*-codes and we provide details about their properties. In Section IV, we present the results of the *I*-codes implementation. In Section V, we analyze robustness of *I*-codes to interference. In Section VI, we show various applications of *I*-codes. In Section VII, we present the security analysis of *I*-codes. In Section VIII we describe the related work. Finally, we conclude the paper in Section IX.

## II. PROBLEM STATEMENT AND ASSUMPTIONS

We address the following problem: *Assuming that two entities ( $A$  and  $B$ ) share a common (radio) communication channel, but do not share any secrets or authentication material (e.g., shared keys or authenticated public keys), how can the messages exchanged between these entities be authenticated and how can their integrity be preserved in the presence of an attacker ( $M$ )?* Here, by message integrity, we mean that the message must be protected against any malicious modification, and by message authentication we mean that it should be clear who the sender of the message is.

We assume that the two entities involved in the communication ( $A$  and  $B$ ) do trust each other; otherwise, little can be done. Whenever we speak of the security of a given protocol, we implicitly assume that the entities involved in the protocol are not compromised. We do assume that the entities know the (public) protocol parameters.

We adopt the following attacker model. We assume that the attacker Mallory ( $M$ ) controls the communication channel in a sense that he can eavesdrop messages and modify transmitted messages by adding his own messages to the channel. We further assume that the attacker cannot disable the communication channel (e.g., use a Faraday's cage to block the propagation of radio signals) between  $A$  and  $B$ . The attacker can jam the transmission and in that way prevent the transmission of the information contained in the message. However, the receiver will still receive the message from the sender, superimposed by the attacker's messages. Finally, we assume  $M$  to be *computationally bounded*.

It is interesting to observe that the security of *I*-codes themselves does not depend on the attacker being computationally bounded. However, authentication schemes derived from *I*-codes presented in Section VI do require the attacker to be computationally bounded.

Our attacker model is similar to the the Dolev-Yao model in that the attacker controls the communication channel, but it differs in that we assume that the attacker cannot fully schedule message transmission as it cannot disable the communication channel. This means that the attacker cannot trivially remove the energy of the signal from the channel (we discussed this in more detail in [33]).

Before introducing our solution to the above stated problem, we give some examples of attacks on message integrity on the radio channel, which are relevant to our proposal. Figure 1 shows two types of such attacks. The first type of attack is called *bit flipping*, in which the attacker introduces a signal on the channel that converts bit "0" into "1" or vice-versa. This attack is shown in Figure 1(a) and Figure 1(b) for messages modulated using amplitude and frequency modulation, respectively. Here, the bit is flipped such that the attacker adds to the channel the signal of the opposite phase to the one representing the bit and the signal representing the opposite bit. The second type of attack is the signal *overshadowing attack*, shown in Figure 1(c). In this attack, the attacker adds to the channel a signal representing a bit string different from the one sent by the honest entity with a significantly higher power than the one of the original signal. In this way, the original signal, regardless of its format or modulation, becomes entirely overshadowed by the attacker's signal, and is treated as noise by the receiver.

In the following sections, we show how these and similar attacks on message integrity can be detected through the use of *I*-codes in conjunction with on-off keying and signal anti-blocking components. Even though we make a clear distinction between *I*-codes and on-off keying, that is, signal anti-blocking, we will often abuse the terminology and call the triple (*I*-codes, on-off keying, signal anti-blocking) an *I*-code.

## III. INTEGRITY (*I*)-CODES

In a way similar to a message authentication code (MAC) and to a signature scheme, integrity codes (*I*-code) provide a method of ensuring the integrity and authentication of a message transmitted over a public channel. The main difference is that *I*-codes remove the assumption that the parties involved in the message exchange share some prior secrets or/and certified public keys.

*I*-codes allow a receiver  $B$  to verify the integrity of the message received from the sender  $A$ , based solely on message coding. *I*-codes consist of three components: unidirectional message coding, on-off keying communication and receiver's awareness of presence in the sender's transmission range. *Unidirectional Error-Detecting* codes [7], [9], [18], [8] are used in situations where it is possible to change, for example, a bit "0" into a bit "1" but the contrary is not possible (except with a negligible probability). A unidirectional error-detecting code is able to detect any number of unidirectional errors in the given codeword; in other words, for a given error-detection code, no unidirectional error can transform a (valid) codeword into another (valid) codeword. On-off keying is a modulation by which the bit "1" is transmitted on the channel as the *presence* of a signal and the bit "0" is transmitted as the *absence* of a signal. Signal anti-blocking means that the energy of the signal (bit "1") cannot be annihilated by an adversary (we will show several ways how to ensure this). On-off keying therefore creates a modulation scheme supporting the usage of unidirectional error-correcting codes. We demonstrate the operation of *I*-codes through an example of message transmission over a radio

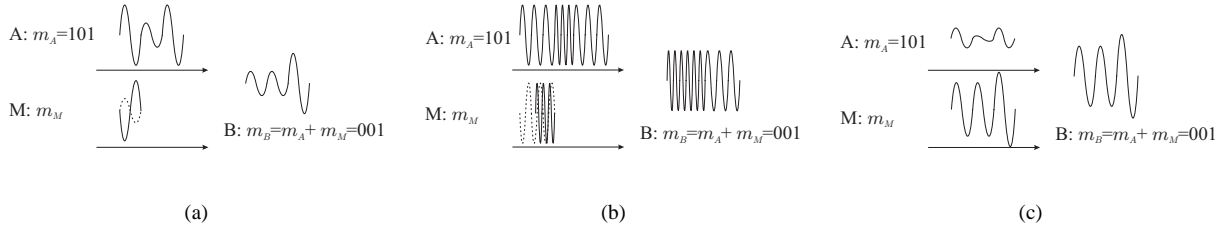


Fig. 1. Example of attacks on message integrity: (a) *Bit flipping; signals modulated using amplitude modulation (AM)*; (b) *Bit flipping; signals modulated using frequency modulation (FM)*; (c) *Signal overshadowing; signals modulated using amplitude modulation*.

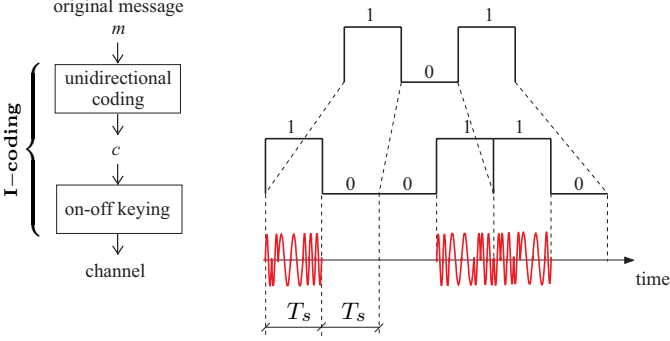


Fig. 2. *I-coding: An example of I-coding at the sender using unidirectional Manchester encoding: 1  $\rightarrow$  10 and 0  $\rightarrow$  01.*

channel, shown on Figure 2. In this example, a message  $m$  is first encoded (to code  $c$ ) using unidirectional Manchester codes and is then transmitted over the radio channel using on-off keying. Each bit “1” of  $c$  is transmitted as a random and freshly generated signal of duration  $T_s$  (the *symbol period*), and each bit “0” as the absence of signal of the same duration. The signals representing bit “1” do not carry any information, but it is the *presence* or *absence* of their energy in a given time slot of duration  $T_s$  that conveys information<sup>1</sup>. Manchester coding encodes any message  $m$  by transforming each bit “1” into 10 and each bit “0” into 01. This encoding is injective and will result in a code  $c$  with the same number of bits “1” and “0”, i.e., the Hamming weight  $H(c)$  of the code will be equal to the size of the original message ( $H(c) = |m|$ ).

In order to retrieve the transmitted codeword, the receiver measures the energy in the corresponding time slots of duration  $T_s$ . Here, we assume that the sender and the receiver can synchronize at the physical layer and with respect to the beginning and the end of the transmission of  $c$ ; in Section III-A, we show how this can be simply achieved. Let  $P_r$  denote the average power that the receiver measures in a time slot of duration  $T_s$ . Let us also denote with  $P_0$  and  $P_1$  pre-defined *threshold power levels*. Here,  $P_1 \geq P_0$ . For the considered time slot, the receiver decodes the received signals as follows:

- if  $P_r \geq P_1$ , output symbol “1”
- if  $P_r \leq P_0$ , output symbol “0”
- else reject.

To verify the integrity and authenticity of the demodulated message  $m$ , the receiver needs to

- 1) verify that it resides in the sender’s coverage area
- 2) verify that the channel on which it received the signal is the channel used by the sender

- 3) verify that the demodulated code  $c$  is valid, i.e., it contains an equal number of symbols “1” and “0”

If these conditions are fulfilled, the receiver concludes that the received message  $m$  is authentic and has been transmitted by the sender. Conditions (1) and (2) are generally fulfilled by dissemination of public information; namely, the area that the sender covers and the channels that the stations use can be made publicly available (or disseminated) prior to the start of communication. This especially applies to broadcast authentication scenarios, where the senders are permanently present in a given area and broadcast security-critical information. Condition (3) therefore is the most important criterion for the verification of message authenticity and integrity. This condition ensures that if the Manchester code  $c$  contains an equal number of symbols “1” and “0”, then it has not been modified in transmission. This is due to the on-off keying modulation and signal anti-blocking property which prevent “1” symbols from being flipped, and enable the detection of signal overshadowing attacks. For example, to convert  $c = 100110$  into  $c' = 101010$ , the fourth bit of  $c$  has to be changed to “0”. This, the attacker cannot do since each bit “1” is transmitted as a freshly generated random signal and its energy therefore cannot be annihilated from the channel (except with negligible probability; for more detail see Section VII). Finally, if the received codeword is valid, the receiver uses Manchester decoding (i.e., 01  $\rightarrow$  0, 10  $\rightarrow$  1) to retrieve the emitted message  $m = 101$ .

We note that the adversary can still convert any symbol “0” to symbol “1”. In this case, however, the receiver will simply drop the received codeword, since such a codeword cannot be decoded properly. Assume that the adversary flips the third symbol “0” into symbol “1” in the original codeword  $c = 100110$ . The receiver will try to decode the altered codeword as 101110. This codeword cannot be related to any message, since there is no transformation defined for the bit-pair 11. The received code will therefore be rejected by the receiver.

Note that the receiver does not have to know the shape of (random) waveforms emitted by the sender. All the receiver has to know is the frequency band used by the sender; in this sense, the receiver can be viewed of as simple radiometer measuring the energy in a given frequency band.

#### A. Synchronization

So far, we have assumed that the sender and the receiver are synchronized with respect to the beginning and the end of the transmission of a given codeword  $c$ . In this section, we show how this can be achieved.

In the case that all messages transmitted by the sender have the same size, the synchronization between the sender and the receiver is trivial; it is sufficient that the size of the messages

<sup>1</sup>Note that this is similar to the *pulse position modulation (PPM)*.

(and implicitly of the codewords) is publicly known and that the receiver knows when the sender is transmitting on a given channel (not necessarily when the sender starts transmitting, but simply that the sender is transmitting at the time of message reception). This synchronization step allows the receiver to verify if the message is authentic and if it was not modified in transmission. In broadcast authentication and secure navigation applications (Sections VI-A and VI-C), the sender transmits continuously, therefore allowing the receivers to implicitly synchronize as soon as they start receiving sender's signals.

If the transmitted messages are of variable (and unpredictable) sizes, a more elaborate synchronization scheme is required. In what follows, we show a synchronization scheme devised for  $I$ -codes based on Manchester message coding. Our scheme is based on message "delimiters", that enable any receiver to recognize the beginning and the end of each message. Let us assume that the sender wants to transmit the following codeword  $c = 1010011001$  (which corresponds to the message  $m = 11010$  under Manchester encoding). In this case, the sender will simply continuously emit (using the on-off keying) the following repetitive sequence

$$\dots \text{delimiter} \overbrace{1010011001}^c \text{delimiter} \overbrace{1010011001}^c \text{delimiter} \dots \quad (1)$$

Here, "delimiter" represents a specially constructed bit string such that any *successfully demodulated codeword*<sup>2</sup> received between any two consecutive "delimiters" is authentic (i.e., corresponds to 1010011001 in our example). We will show shortly how to construct such a delimiter for the complementary encoding rule.

The receiver first has to make sure that the peer sender is active (transmitting the above repetitive sequence); this step can be omitted if the transmitter permanently emits (e.g., navigation) messages. Then it decodes a codeword received between any two consecutive "delimiters". If the codeword can be converted back to a message using Manchester decoding (i.e.,  $(10 \rightarrow 1, 01 \rightarrow 0)$ ), the receiver accepts this message as being authentic. A nice property of this approach is that the receiver does not have to know the length of the codeword being transmitted in advance.

We next define more formally the notion of "delimiter". Then we construct the delimiter for the complementary encoding rule.

*Definition 1:* For the fixed set of codewords  $\mathcal{C}$ , we define an incongruous delimiter (shortly,  $i$ -delimiter) to be a finite minimum-length string of bits that satisfies the following conditions:

- No substring (of consecutive bits) of any codeword  $c \in \mathcal{C}$  can be converted into the  $i$ -delimiter, without flipping at least one bit "1" of  $c$  to bit "0";
- The  $i$ -delimiter cannot be converted into a substring (of consecutive bits) of any  $c \in \mathcal{C}$ , without flipping at least one bit "1" of the  $i$ -delimiter to bit "0";
- Any valid codeword (i.e., any  $c \in \mathcal{C}$ ) received between two consecutive  $i$ -delimiters is authentic.

*Example 1:* Here,  $\mathcal{C}$  is a finite set of binary codewords, derived using Manchester coding, from a set  $\mathcal{S}$  of all possible binary messages (of size  $|m|$ ).

Consider the set  $\mathcal{C}$  such that  $c = 10100110 \in \mathcal{C}$ . Consider also the following candidate for the  $i$ -delimiter:  $x = 11011$ . We will show that bit-string  $x$  does not satisfy Definition 1 and therefore is not an  $i$ -delimiter for the set  $\mathcal{C}$ . This is easily seen by observing

<sup>2</sup>In our example, by "successfully demodulated codeword" we mean the codeword for which the transformation  $(10 \rightarrow 1, 01 \rightarrow 0)$  exists.

that  $10100110 \rightarrow 10110110$ , i.e., it is sufficient to flip only the fourth bit of  $c$  from "0" to "1" so that  $x$  emerges as the substring of  $c$ . Therefore, the first condition of Definition 1 is not met.

Assuming that an adversary cannot flip bit "1" into bit "0", we have the following result.

*Theorem 1:* Consider the set of codewords  $\mathcal{C}$  obtained by applying the encoding rule  $(1 \rightarrow 10, 0 \rightarrow 01)$  to the set of source states (messages)  $\mathcal{S} = \{0, 1, 00, 01, \dots, \overbrace{11 \dots 1}^k\}$ , for arbitrary  $k < \infty$ . String 111000 is an  $i$ -delimiter for the set  $\mathcal{C}$ .

*Proof:* By mere inspection of all the strings of a length smaller than 6 bits, it easily follows that no such string satisfies Definition 1.

Consider now the string 111000. Observe that for every codeword  $c \in \mathcal{C}$  the number of consecutive bits 0 and the number of consecutive bits 1 is at most two. Therefore, (i) 111000 cannot be converted into any codeword  $c \in \mathcal{C}$  without flipping at least one of the leading bits "1" in 111000 to bit "0", and (ii) no substring of any codeword  $c \in \mathcal{C}$  can be converted into 111000, without flipping at least one bit "1" of  $c$  to bit "0". Thus, the string 111000 satisfies the first two conditions in Definition 1.

We next show that it satisfies the third condition as well. We observe that it is sufficient to focus on a codeword between two consecutive strings 111000, since three consecutive bits "1" never appear in any valid codeword from  $\mathcal{C}$  and the adversary cannot flip a bit "1". Let us consider the following sequence of bits for any  $k$ -bit codeword ( $k$  being even)  $c = (c_1 c_2 \dots c_{k-1} c_k) \in \mathcal{C}$

$$\dots 111000 c_1 c_2 \dots c_{k-1} c_k 111000 \dots \quad (2)$$

We first show that the adversary cannot accomplish that the string 111000 emerges in any (other) part of the sequence (2) and that at the same time any resulting codeword  $\hat{c}$  is valid. As the result the only hope for the adversary is to leave the original delimiters 111000 intact and try to transform the original codeword  $c$  into a different codeword  $\hat{c}$  of the same length. Since  $c$  is an  $I$ -code codeword, the adversary would have to flip at least one bit "1" of  $c$  into a bit "0". However, by assumption he cannot accomplish this.

We now prove that the adversary cannot achieve that the 111000 sequence emerges in any (other) part of the sequence (2) and that at the same time any resulting codeword  $\hat{c}$  is valid. For this, let us consider all possible 6-bit substrings (of consecutive bits) in the sequence (2). These can be captured by one of the eleven cases given below:

1.  $1 \boxed{11000 c_1} c_2 \dots c_{k-1} c_k 111000$
2.  $11 \boxed{1000 c_1 c_2} c_3 \dots c_{k-1} c_k 111000$
3.  $111 \boxed{000 c_1 c_2 c_3} c_4 \dots c_{k-1} c_k 111000$

4. 1110  $\overbrace{00c_1c_2c_3c_4}^c c_5 \dots c_{k-1}c_k 111000$
5. 11100  $\overbrace{0c_1c_2c_3c_4c_5}^c c_6 \dots c_{k-1}c_k 111000$
6. 111000  $\dots c_{i-4} \overbrace{c_{i-3}c_{i-2}c_{i-1}c_i c_{i+1}c_{i+2}}^c \dots 111000$
7. 111000  $c_1c_2 \dots c_{k-5} \overbrace{c_{k-4}c_{k-3}c_{k-2}c_{k-1}c_k 1}^c 111000$
8. 111000  $c_1c_2 \dots c_{k-4} \overbrace{c_{k-3}c_{k-2}c_{k-1}c_k 11}^c 1000$
9. 111000  $c_1c_2 \dots c_{k-3} \overbrace{c_{k-2}c_{k-1}c_k 111}^c 000$
10. 111000  $c_1c_2 \dots c_{k-2} \overbrace{c_{k-1}c_k 1110}^c 00$
11. 111000  $c_1c_2 \dots c_{k-1} \overbrace{c_k 11100}^c 0$

*Case 2 – Case 5.* The strings  $(1000c_1c_2)$ ,  $(000c_1c_2c_3)$ ,  $(00c_1c_2c_3c_4)$  and  $(0c_1c_2c_3c_4c_5)$  cannot be transformed into the string 111000 without flipping at least one bit “1”, since  $c_1 \oplus c_2 = 1$  and  $c_3 \oplus c_4 = 1$  (by the complementary encoding).

*Case 6.* We showed at the beginning of the proof that the string 111000 satisfies the condition one in Definition 1. So no string  $(c_{i-3}c_{i-2}c_{i-1}c_i c_{i+1}c_{i+2})$ ,  $i \in [4, 5, \dots, k-2]$ , can be transformed into the string 111000 without flipping at least one bit “1”.

*Case 7 – Case 11.* The strings  $(c_{k-4}c_{k-3}c_{k-2}c_{k-1}c_k 1)$ ,  $(c_{k-3}c_{k-2}c_{k-1}c_k 11)$ ,  $(c_{k-2}c_{k-1}c_k 111)$ ,  $(c_{k-1}c_k 1110)$  and  $(c_k 11100)$  cannot be converted into the string 111000 without flipping at least one bit “1”, since they all contain at least one bit “1” among the last three digits.

*Case 1.* The string  $(11000c_1)$  can be transformed into the string 111000 by flipping the third bit to “1”, conditioned on  $c_1 = 0$ . In this case, the bit  $c_2 = 1$  becomes the first bit of the new codeword  $\hat{c}$  (not necessarily valid). From *Case 2* to *Case 11* above we know that the ending of the codeword  $\hat{c}$  must be denoted either by the original delimiter 111000 or by the delimiter obtained by joining the first bit “1” of the original delimiter to the new codeword  $\hat{c}$ . In the first case, the length of the resulting codeword  $\hat{c}$  is  $k-1$  (an odd number) and so  $\hat{c}$  cannot be a valid codeword. In the second case, one bit “1” is added to the sequence that already has a deficit of bits “0” (i.e., the bit  $c_1 = 0$  is not a part of  $\hat{c}$ ) and so the resulting codeword  $\hat{c}$  cannot be not valid.

We conclude the proof by observing that the string 111000 is the shortest string (i.e., 6 bits long) that satisfies all the conditions in Definition 1. ■

*Remark 1:* It is interesting to observe that for the complementary encoding rule and the delimiter 111000, the first two conditions from Definition 1 imply the third one (they are sufficient). Whether this holds in general (for any  $I$ -code and an  $i$ -delimiter) is an interesting open problem.

Referring back to the example (1), the sender can preserve the integrity of message 11010 (i.e., the codeword  $c = 1010011001$ ) by simply emitting (using the on-off keying) the following repetitive sequence

$$\dots \underbrace{111000}_{i\text{-delimiter}} \overbrace{1010011001}^c \underbrace{111000}_{i\text{-delimiter}} \overbrace{1010011001}^c \underbrace{111000}_{i\text{-delimiter}} \dots$$

The receiver decodes a codeword received between any two consecutive  $i$ -delimiters (after having verified that the peer sender is active). Any successfully demodulated codeword between two  $i$ -delimiters must have been emitted by the peer sender (the codeword is authentic). At this stage, the peer sender can stop transmitting the above repeated sequence. The important impli-

cation of the synchronization based on  $i$ -delimiters is that the receiver does not have to know in advance the length of the message to be transmitted by the sender.

## B. Unidirectional Error Detecting Codes

So far, we have assumed that the messages emitted by the sender are encoded using Manchester coding. As we already noted, Manchester coding is only one possible way on message encoding in  $I$ -codes. In fact, practically any unidirectional error detecting codes [7], [9], [18], [8], [31] can be used instead of Manchester coding to support  $I$ -codes.

By definition, a unidirectional code  $\mathcal{C}$  detects all unidirectional errors if and only if every pair of codewords in  $\mathcal{C}$  are unordered. Two codewords  $c$  and  $c'$  of size  $n = |c| = |c'|$  are ordered if either  $c[i] \geq c'[i]$  or  $c[i] \leq c'[i]$ ,  $\forall i \in 1, 2, \dots, n$ ; they are unordered otherwise. A unidirectional error (in our scenario  $0 \rightarrow 1$ ) therefore always transforms a codeword  $c$  to an invalid codeword  $c' \notin \mathcal{C}$  which is either smaller or greater than  $c$ .

Considering the size of the message check value, Manchester codes are not optimal detection codes; for each message  $m$ , Manchester coding produces a code of size  $|2m|$ , which means that the size of the message check value is equal to the size of the original message  $|m|$ . Although Manchester codes have other desirable properties besides error detection, if check value size is the only criteria for code selection, Manchester codes can be replaced by Berger codes [27], which are optimal in terms of the check value size of available separable unordered codes. Berger codes append  $k = \lceil \log_2(|m|+1) \rceil$  check bits to a message of size  $|m|$ . The message check value is created such that the number of “1”s are counted in the  $|m|$  message bits. The resulting binary number is complemented and appended to the message bits. For example, if a message  $m = 1001101$  is to be transmitted, then the size of the check value will be  $\lceil \log_2(7+1) \rceil = 3$ , and the check value will be  $k = \overline{100} = 011$ . The transmitted code will therefore be  $c = 1001101011$ .

In case when transmitted messages have a fixed and publicly known sizes, Berger codes clearly outperform Manchester coding in term of check value communication overhead. However, if transmitted messages have unpredictable sizes, appropriate delimiters need to be developed for Berger codes. As we showed in the previous section, the optimal  $i$ -delimiter for Manchester coding is a six-bit sequence 111000. Note that the size of the  $i$ -delimiter does not depend on the size of the message that is being transmitted. Thus, we simply compute the per-message overhead of  $I$ -coded message using Manchester coding as  $|m| + 6$ .

If  $I$ -codes are based on Berger codes, or on similar codes that append check values to (arbitrary) messages, the size of  $i$ -delimiters will correspond to the size of the largest message size plus the size of the corresponding check value. This is the consequence of the fact that transmitted messages do not have any particular structure (as is the case of Manchester coding), and a smaller message could therefore always be seen as a part of a larger message. For Berger codes, an  $i$ -delimiter can thus be constructed as a message 0, with corresponding check value (containing only bits “1”. Similarly, appropriate delimiters can be devised for other coding scheme; this we leave for future investigations. Finally, the choice of a unidirectional detection code on which  $I$ -codes are based depends mainly on the application in which  $I$ -codes are used, message sizes, and on the limitations of the systems on which they are implemented.



Fig. 3. Implementation of  $I$ -codes based on Atmel AT86RF211 transceiver.

In the following sections, we report on our experience with the real-life implementation of  $I$ -codes and we introduce the novel concept of *authentication through presence*.

#### IV. IMPLEMENTATION OF $I$ -CODES

In this section, we provide results of our  $I$ -codes implementation feasibility study.

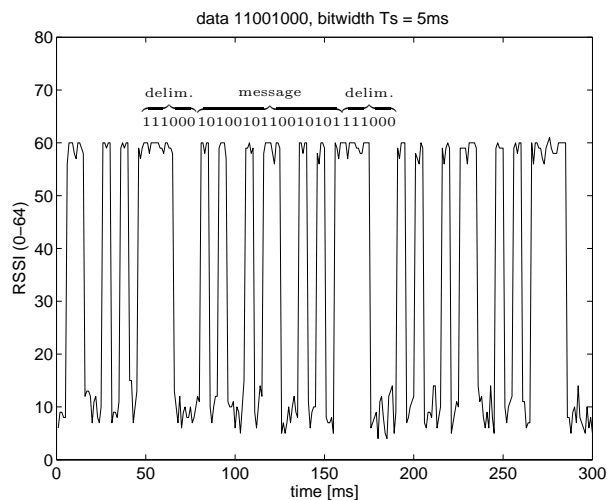
##### A. Implementation with Atmel Radio Transceiver

We implemented  $I$ -codes using Atmel AT86RF211 (aka: TRX01) single chip radio transceiver [1]. TRX01 chip is optimized for licence-free ISM band operations from 400 MHz to 950 MHz and uses FSK modulation spectrum shaping. It has programmable output power (0 to 13 dBm) and the RSSI dynamic range of 50 dB (from -95 dBm to -45 dBm RF input signal power). The corresponding RSSI code ranges from 0 in steps by 1 to maximum 64 (for RF input signal power  $\geq$  -45 dBm). In our implementation, the transceiver works in the 433MHz frequency band (using one out of 64 available channels) and is mounted on a supporting daughter-board that is equipped with a serial port, power-supply connectors and the antenna (see Figure 3). With Atmel AT86RF211, each RSSI reading takes a bit less than 1 ms; note that this directly affects the duration of  $I$ -code bits 1.

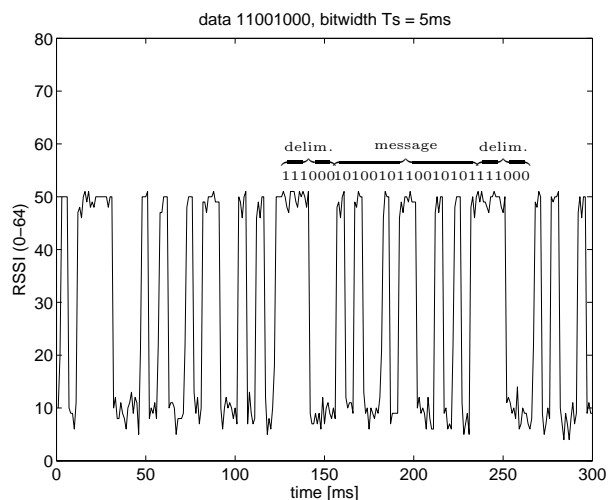
In our  $I$ -code implementation, each original message  $m$  is first Manchester coded such that each “1” is transformed into a “10” and “0” into a “01”. An encoded message is then transmitted such that each “1” is transmitted as an FSK modulated signal carrying a random data and each “0” is transmitted as an absence of signal of duration  $T_s$  (we can vary  $T_s$  in the range from 1 to 100 ms). A message transmitted can be of an arbitrary (finite) size. The beginning and the end of each message is marked by the  $i$ -delimiter 111000.

The decoding process at the receiver is implemented as follows. A “silence period” on the channel of the duration of  $T_s$  ms (e.g., we used 5 ms in our experiments) is interpreted as a “0”, whereas the presence of a signal is interpreted as “1”. Here, the “silence on the channel” is defined as a period during which the received signal strength (RSSI) on the receiver remains below a preset RSSI level. Otherwise, the signal is interpreted as “1”.

We performed a set of experiments with our implementation. Figures 4(a) and 4(b) show two RSSI traces for the (line-of-sight) transmission of message 11001000 (i.e., Manchester encoded 1010010110010101) between two  $I$ -codes devices that are separated by 1.5 and 12 m. The duration  $T_s$  of each bit is set



(a) The sender and the receiver are around 1.5 m apart.



(b) The sender and the receiver are around 12 m apart.

Fig. 4.  $I$ -codes trace (RSSI) for the (line-of-sight) transmission of message 11001000 and different distances between the sender and the receiver.

to 5 ms thus giving the throughput of 100 bps. By further reducing  $T_s$ , we can achieve even faster communication. For example, a novel Atmel transceiver can take RSSI readings in 0.5 ms which for  $T_s = 2ms$  (4 RSSI readings per bit) results in the throughput of 250 bps. This is more than sufficient data rate for our purposes (e.g., transmission of hash values, short authentication strings, etc.) as we explain in Section VI.

We further studied the impact of an interferer (attacker) on the ongoing transmission. Our goal is twofold: (i) show the impact of an interferer on the ongoing transmission, and (ii) verify that an attacker cannot flip a bit “1” to bit “0” (by merely emitting on the same frequency). A thorough analysis of  $I$ -codes from the signal cancellation point of view is presented in Section VII. From Figure 5 we can see that it is, as expected, fairly easy to jam  $I$ -coded message (as any radio signal where the transmitter and the receiver do not share any secret like hopping sequence or spreading code). However, compared to other systems, it is much easier to detect such a jamming activity in the context of  $I$ -codes. More importantly, Figure 5 shows that the interferer have not had been able to cancel a radio signal (to flip a bit “1” into a bit “0”), throughout the observation period (couple of hours

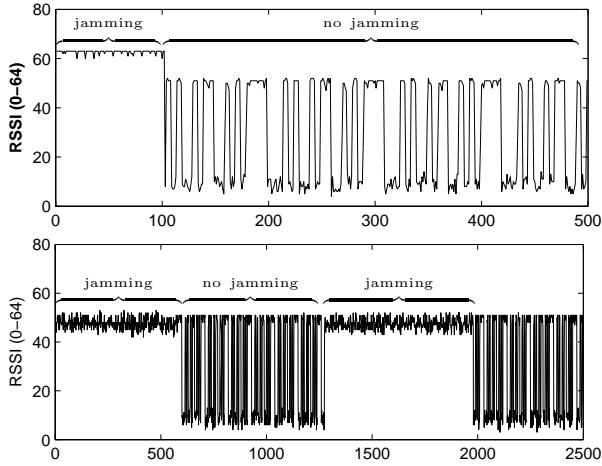


Fig. 5.  $I$ -codes trace (RSSI) for message 11001000 with an interferer (attacker) who is jamming the communication (the distance between the “legal” transmitter and the receiver is around 12 m): the top figure corresponds to the scenario where the attacker is located in the close vicinity of the receiver and in the bottom figure the attacker is collocated with the “legal” transmitter.

long - the figure shows RSSI trace for only a small part of the observation period). In Section VII, we analyze this aspect more thoroughly.

### B. Implementation on wireless sensor platform

We further implemented  $I$ -codes on Mica2 sensor networking platform [3]. This platform consists of a processor and a CC1000 radio. CC1000 is a single-chip RF transceiver, has a programmable frequency (300-1000 MHz), and uses FSK modulation spectrum shaping. It has programmable output power, (-20 to 10 dBm) and a high receiver sensitivity (-110 dBm).

In Mica2 implementation of  $I$ -codes, we use pairs of sensors running the SOS operating system [15]. Each original message  $m$  is first  $I$ -coded such that each “1” is transformed into a “10” and “0” into a “01”. An  $I$ -coded message is then transmitted such that each “1” is transmitted as an SOS packet containing a random payload of length  $k$  (the payload is chosen randomly for each packet) and each “0” is transmitted as an absence of signal of duration  $T_s$  (in our implementation the number of chips per symbol “1” is  $k = 48$  bits and  $T_s = 10$  ms). Each packet consists of a preamble and of a payload. The preamble is 12 bytes long and with the payload makes a total of 18 bytes per packet. The decoding process is similar to decoding in Atmel implementation.

### C. Implementation using 802.11 wireless cards

In [26], we presented our implementation of  $I$ -codes using 802.11 wireless cards. In this experiment, the signals were sent using a standard built-in Atheros 5212 IEEE 802.11 wireless network card and received by a software radio [1]. Both the transmitter and the receiver were equipped with built-in omnidirectional antennas, whose gains were not enhanced. The transmission power at the sender was set to 100mW. Our measurements show that the receiver can clearly distinguish symbol “1” from environmental noise (i.e., symbol “0”) up to almost 100m. Note that these results can be further enhanced if higher transmission powers and antenna gains are used at the stations. Besides the reach of signals, our experiments also included the estimation of

the maximal rate of  $I$ -coded signals. Using available MadWifi [2] drivers, we were able to transmit  $I$ -coded signals with bit durations of approx. 2 ms (i.e., symbol durations of approx. 1 ms). This experiment showed that with off-the-shelf cards and drivers, the data rate of  $I$ -coded signals can be around 500 bits/s. Appropriate modifications of the wireless card drivers of the sender (and receiver) will allow the rate of the  $I$ -coded signals to be further increased; this is part of our ongoing work.

## V. ROBUSTNESS

In this section, we analyze the robustness of  $I$ -codes to jamming and interference (intentional and unintentional). Because  $I$ -codes use on-off keying, they are generally not robust to continuous interference or jamming (Figure 5).

Recall, if the receiver receives sufficient signal power within a predefined time slot (of duration  $T_s$ ), it will interpret it as the symbol “1”. This means that any interference whose average power within a time slot  $T$  is above a predefined threshold  $P_1$  will be also interpreted as the symbol “1” at the receiver. Continuous interference or jamming will therefore turn each  $I$ -coded message into an unusable stream of “1” symbols (Figure 5). However, if Manchester coding is used as an unidirectional code,  $I$ -coded messages still provide robustness to casual interference or jamming, and enable message reconstruction. This follows from the fact that **each bit “0” is a correct bit** - by assumption (and construction of  $I$ -codes) an attacker cannot annihilate the radio signal.

To exemplify this, let us consider a possible usage of  $I$ -codes as shown in Figure 7. Thus, the original message  $m$  is sent (e.g., using an IEEE 802.11 card) over an insecure high-bandwidth radio channel  $C_1$ , whereas the  $I$ -coded hash value  $h(m)$  of  $m$  is sent on a different channel  $C_2$  (in Section VI we discuss this approach in greater detail). Now, if due to interference, the  $I$ -coded message ...100110... is received at the receiver as ...101110..., the receiver will be able to detect that the error occurred in symbol three or four (i.e., a symbol string 11 cannot be decoded into a valid bit using Manchester coding). Upon detection, the receiver can reconstruct the message by turning symbols three and four into 01 and 10 and observing which of the messages (...101010... or ...100110...) corresponds to the hash value of the message received over channel  $C_1$ . Here, on-off keying enables immediate and simple recognition of transmission errors, and Manchester coding enables the receiver to reconstruct the erroneous bits.

Note here that a potential attacker can try to overshadow the original message  $m$  sent over  $C_1$  with his own message  $m'$  and at the same time cause changes to the  $I$ -coded message so that the receiver accepts the forged message as valid (all verifications pass). In order for this attack to be successful, the attacker has to ensure that the receiver reconstructs the  $I$ -coded message as  $h(m')$ . Now the trick is to let the receiver try to reconstruct the  $I$ -coded hash value only if the number of erroneous bits is less than some threshold  $N_e \leq L$ , where  $L$  is the size of  $I$ -coded message (e.g., in our case,  $L = 160$  bits and, for example,  $N_e = 25$ ). In this way, we force the attacker to find a 2nd preimage of the truncated output (in our example, 135 bits long) of the cryptographic hash function. But for a high number of “correct” hash value bits (i.e.,  $L - N_e$ ), this is infeasible for the computationally bounded attacker. Note also that  $N_e$  determines the average effort ( $2^{N_e - 1}$ ) of the receiver in the

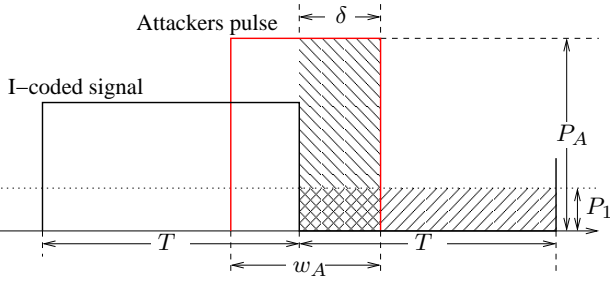


Fig. 6. Visualization of equation (3). The signal marked “Attackers pulse” is the attackers jamming pulse with width  $w_A$  and height  $P_A$ . Jamming is only successful in the shaded area  $\delta \cdot P_A$  is bigger than, or equal to, the shaded area  $T \cdot P_1$ .

message reconstruction procedure. In a non-adversarial setting, the above approach to increasing robustness allows  $I$ -codes to operate in noisy (realistic) environments.

If interference or jamming appears in random bursts, those bursts might overlap with a part of the signal that was transmitted as a “1” and thus cause no interference. The probability of successful jamming (or harmful interference) depends on the length of the jamming burst and the length of the time slots (i.e., the symbol-width) of the  $I$ -coded signal. The jamming burst scenario is shown in Figure 6 and the probability of successful random jamming (interference) is given by:

$$Pr_{\text{interf}} = \begin{cases} 0 & \text{for } w_A < \delta \\ \frac{T+w_A-2\delta}{2T} & \text{for } \delta \leq w_A \leq T+2\delta \\ 1 & \text{for } w_A > T+2\delta \end{cases} \quad (3)$$

where  $T$  it the width of the time slot,  $w_A$  it the width of the attacker’s pulse and  $\delta$  is the width of the pulse required to change a 0 to a 1.  $\delta$  is a function of the threshold  $P_1$ , the time slot width  $T$  and the power of the attackers transmission  $P_A$ :

$$\delta = \frac{P_1 T}{P_A} \quad (4)$$

As we can see from (3), narrow pulses of interference (less than  $\delta$ ) are simply ignored. Equation (4) shows that there are two ways a  $I$ -coded message can be made more resistant to jamming. The first is increasing the threshold  $P_1$  to force the attacker to either use more power or “hit” the zeros in the  $I$ -coded transmission more accurately. The second way to increase robustness is to increase the width of the time slot  $T$ . To see this more clearly we can rewrite (3) and (4) to:

$$Pr_{\text{interf}} = \frac{1}{2} + \frac{w_A}{2T} - \frac{P_1}{P_A} \quad \text{for } TP_1 \leq P_A w_A \leq T(2P_1+1) \quad (5)$$

Here it is clear that a bigger  $T$  forces the attacker to spend more power, however, a larger time slot has the unwanted consequence of lowering the bit rate. The optimal setting for these parameters will depend on the application scenario and the attacker model. Regardless of the values of the above mentioned parameters, the receiver will detect any successful jamming or interference with 100% probability. Thus, while the attacker can jam the system, he cannot do so without being detected.

## VI. AUTHENTICATION THROUGH PRESENCE

Using  $I$ -codes, we develop a novel concept called *authentication through presence*, which enables (broadcast) message authentication based solely on the awareness of the presence in

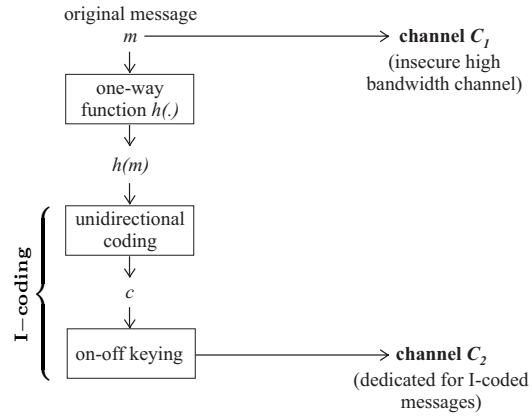


Fig. 7. Possible usage of  $I$ -codes for integrity protection: Original message is transmitted over an insecure high-bandwidth channel  $C_1$ , whereas the integrity protection is enabled with  $I$ -codes on a different channel  $C_2$ .

the power range of an entity. We first introduce this concept and then we describe its use in two application scenarios: broadcast authentication and key establishment.

We describe our concept through an example involving two parties: the sender  $A$  and the receiver  $B$ . Note that the sender and the receiver do not share *any* authentication material. The main idea of our approach is shown in Figure 7. The message  $m$ , whose integrity needs to be protected, is sent over a channel  $C_1$  which does not protect its integrity and over which its authenticity cannot be verified. This channel can be realized as *any* communication channel. The message digest  $h(m)$  (e.g., the message hash) is sent over a separate communication channel  $C_2$ , dedicated for integrity protection (we have shown through our implementation in Section IV that this dedicated channel can be realized using existing communication channels). Thus, if  $A$  wants to send a message to  $B$ , she will use the protocol shown in Figure 8.

In this protocol,  $h(\cdot)$  represents a one-way function used to protect the integrity of the transmitted message. This function can be implemented as a simple hash.  $I\text{-code}(h(m))$  represents the  $I$ -coded message digest  $h(m)$ . The sequences preceding and following after  $I\text{-code}(h(m))$  are  $i$ -delimiters (Section III-A), which ensure that the receiver knows the beginning and the end of the  $I$ -coded message.

In this protocol, the integrity and the authenticity of the message  $m$  is verified through the verification of the authenticity and integrity of its digest  $h(m)$ . The authenticity and the integrity of  $h(m)$  is guaranteed with  $I$ -codes if and only if the following conditions are met: (i) the receiver  $B$  knows that it is in the power range of the sender  $A$ , (ii) the receiver  $B$  knows that  $A$  has started transmitting on the integrity channel ( $C_2$ ). The first condition is *the condition of presence* which ensures that the receiver is receiving signals from the sender. The second condition is *the condition of synchronization* which ensures that the receiver knows at which time is the transmission of data performed. If the receiver wrongly believes that the transmitter is transmitting, or if it wrongly believes to be in the power range of the sender, a (malicious) entity can insert false data on the channel and these data will be accepted as valid by the receiver. This follows from the properties of  $I$ -codes, which assume the presence of the signal from the legitimate sender on the channel.

*Remark 2:* The protocol shown in Figure 8 has the drawback that the size of the cryptographic hash function  $h(\cdot)$  increases

$A \rightarrow B$  (on  $C_1$ ) :  $m$   
 $A \rightarrow B$  (on  $C_2$ ) :  $\dots \underbrace{111000}_{i\text{-delimiter}} I\text{-code}(h(m)) \underbrace{111000}_{i\text{-delimiter}} \dots$   
 $B$  : Verify the integrity and the authenticity of  $h(m)$  using  $I$ -codes.  
 Verify the integrity and the authenticity of  $m$  using  $h(m)$ .

Fig. 8. Authentication through presence. An example of the protocol that enables authentication through the presence property;  $h(\cdot)$  represents a one-way function.

over time, in order to compensate for fast (daily) advance in computational technology and computational power available to an adversary. Today a “target collision-resistant” hash function implies the hash function size of at least 80 bits [19]. However, according to [19], the minimum required size increases linearly over time due to fast technological advances. It is not advisable to go below these minimum sizes in the adversarial model where the  $I$ -coded message in Figure 8 can be delayed sufficiently.

Consequently, straightforward solutions similar to the one given in Figure 8 are said to be “time-variant”, that is, the number of bits to be transmitted using  $I$ -codes increases over time. This is clearly not desirable in our context, since the reliability of  $I$ -codes drops quickly with the size of messages. In Section VI-B, we will describe the protocol (in the context of user-friendly key establishment) that allows us to optimally trade-off the size of messages to be  $I$ -coded with the security, and to significantly decrease the size of  $I$ -coded messages.

In the following three sections, we show in which scenarios the conditions of presence and synchronization are fulfilled and in which, therefore,  $I$ -codes can be used for message authentication and integrity protection.

#### A. Access point authentication

Here, we show that authentication through presence can be a useful tool for the broadcast authentication of messages from fixed access points (AP).

Our scenario is depicted in Figure 9. Here,  $I$ -codes are used by the AP to advertise its public key. This key can be later used to provide authentication and integrity protection of all messages generated by the AP.

This enables any user that comes into the range of the AP to know that the advertised public key of this access point is authentic and belongs to the access point in whose range the user is located. If the user trusts the environment in which the access point is placed (a bank or an office), it will trust all information coming from that access point and will use the public key of the AP to establish a secure connection to the station. Here, it is important that the user knows that the environment in which she is placed is covered by at least one legitimate AP. If this condition is fulfilled, it is of little importance if there are any rogue APs present in this space, as long as the legitimate APs are active.

We assume that the sender (AP) is static. The (conservative) reach of its transmission is known to the receivers. The receivers therefore know before they start receiving the data if they are in the sender’s power range or not; this knowledge is a publicly available information. The receivers also know the integrity channel used by the AP to emit its public key. In the case of,

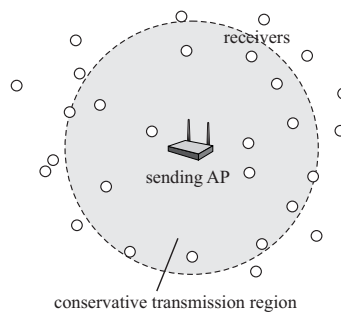


Fig. 9. Broadcast integrity and authentication with an access point. By the “conservative transmission region” we mean the region where the received power of a signal transmitted by the AP exceeds some predefined threshold level (which is a security parameter in our case).

for example, IEEE 802.11a, one of the 12 orthogonal channels can be allocated for this purpose.

The AP continuously sends its key on the integrity channel ( $C_2$  in Figure 7). When it is not advertising its public key, the AP jams the integrity channel to prevent any fake public keys being transmitted over the same channel. As the AP is continuously active, there is no need for synchronization with the receivers; the receivers will start receiving the data when they come into AP’s power range. This power range can be estimated by the user holding a receiver (e.g., the user can estimate that the signal can reach within the room where the AP is located), or can even be marked. Furthermore, to avoid attacks during the time when the AP fails, its status (activity) can be signalled to the receivers through some visual channel (e.g. a blinking LED).

#### B. Key Establishment Over Insecure Channels

In this section we show how authentication through presence can be used for user-friendly key establishment over an (insecure) radio link in peer-to-peer networks. We also show how it is possible to optimally trade-off the security of the key agreement protocol with the size of message to be transmitted using  $I$ -codes. This is particularly important, since  $I$ -codes are not developed with reliability in mind (i.e., a bit zero is conveyed as the absence of a signal).

In [32], we have developed a two-party key agreement protocol that is based on the Diffie-Hellman ( $DH$ ) key agreement protocol.  $DH$  key agreement is known to be vulnerable to the man-in-the-middle attack if the two users involved in the protocol do not share any authenticated information about each other (e.g., public keys, certificates, passwords, shared keys, etc.) prior to the protocol execution. We solve this problem by leveraging on  $I$ -codes that can enable message integrity protection and thus prevent man-in-the-middle attacks.

Our protocol unfolds as shown in Figure 10. Both Alice ( $A$ ) and Bob ( $B$ ) have selected their secret exponents  $X_A$  and  $X_B$ , respectively, randomly from the set  $\{1, 2, \dots, q\}$  ( $q$  being the order of an appropriate multiplicative group) and calculated  $DH$  public parameters  $g^{X_A}$  and  $g^{X_B}$ , respectively.  $A$  and  $B$  proceed by generating  $k$ -bit random strings  $N_A$  and  $N_B$ , respectively. Finally,  $A$  and  $B$  calculate commitment/opening pairs for the concatenations  $0\|ID_A\|g^{X_A}\|N_A$  and  $1\|ID_B\|g^{X_B}\|N_B$ , respectively. Here, 0 and 1 are two public (and fixed) values that are used to prevent a *reflection attack* [21].  $ID_A$  and  $ID_B$  are human readable identifiers belonging to parties  $A$  and  $B$  (e.g., e-mail addresses).

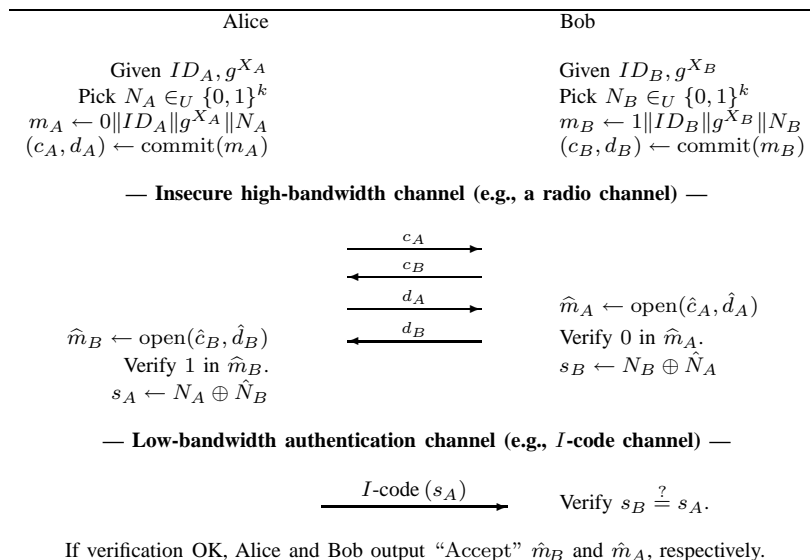


Fig. 10. Diffie-Hellman key agreement protocol based on  $I$ -codes ( $DH^C$ )

The following four messages are exchanged over an insecure (radio) link. In the first message,  $A$  sends to  $B$  the commitment  $c_A$ .  $B$  responds with his own commitment  $c_B$ . In turn,  $A$  sends out  $d_A$ , by which  $A$  opens the commitment  $c_A$ .  $B$  checks the correctness of the commitment/opening pair  $(\hat{c}_A, \hat{d}_A)$  and verifies that 0 appears at the beginning of  $\hat{m}_A$ . If the verification is successful,  $B$  sends, in the fourth message,  $d_B$ , by which  $B$  opens the commitment  $c_B$ .  $A$  in turn checks the commitment and verifies that 1 appears at the beginning of  $\hat{m}_B$ . If this verification is successful,  $A$  and  $B$  proceed to the final phase (Figure 10).

In the final phase,  $A$  and  $B$  first generate the authentication strings  $s_A$  and  $s_B$ , respectively, as shown in Figure 10 ( $\oplus$  is the bitwise “xor” operation). The length of each of these strings is  $k$ . Finally, Alice sends  $s_A$  over the integrity channel to Bob, which then compares it to  $s_B$ . If they match, Alice and Bob accept the DH public keys  $g^{X_B}$  and  $g^{X_A}$ , and the corresponding identifiers  $ID_B$  and  $ID_A$ , respectively, as being authentic. At this stage, Alice and Bob can safely generate the corresponding secret DH key ( $g^{X_A X_B}$ ).

A security analysis of the  $DH^C$  protocol can be found in [32]. Here we only state the result. We denote with  $\gamma$  the maximum number of sessions (successful or abortive) of the  $DH^C$  protocol that any party can participate in. Also, we assume that the used commitment scheme is “ideal”, in the sense that the hiding and binding properties of it always hold.

*Theorem 2 (cf. [32]):* The probability that an attacker succeeds against a targeted user of the  $DH^C$  protocol is bounded by  $\gamma 2^{-k}$ .

Our key agreement protocol exhibits two unique features: (i) it is optimal in the length of the message to be sent using  $I$ -codes, and (ii) it is *time-invariant*, i.e., in spite of the fact that the security parameters of the  $\text{commit}(\cdot)$  function increase over time, in order to compensate for fast (daily) advance in computational technology and computational power available to an adversary [19], the number of bits  $k$  to be transmitted using  $I$ -codes does not change over time (for the fixed security level). Note that that the protocol shown in Figure 8 is not time-invariant, since it relies on the cryptographic hash function  $h(\cdot)$  whose

security parameter increases over time [19].

Let us give an example of possible values for the parameters  $k$  and  $\gamma$ . Let us assume that the fixed user can participate in at most  $\gamma = 2^{20}$  sessions (successful or abortive) in his/her lifetime; this corresponds to 32 sessions per day during approximately 89 years. Then, by choosing  $k = 50$  (bits) we obtain that the highest probability of success by the adversary (having seen a huge number of  $2^{20}$   $DH^C$  sessions by the fixed user) is  $\gamma 2^{-k} = 2^{-30}$ . Note that  $k$  also represents the length of the verification string  $s_A$  to be communicated through  $I$ -codes. Thus, with  $I$ -codes using Manchester encoding and  $T_s = 5$  ms, it takes only 0.5 seconds to transmit 50 bit verification string  $s_A$ . This is rather negligible cost, given that all the messages are transmitted over a radio link.

Therefore, with  $I$ -codes, the involvement of the users in the protocol execution is rather minimal and therefore the  $DH^C$  is indeed user-friendly.

### C. Authentication of navigation signals

In this section, we briefly discuss how integrity-codes can be used to secure navigation signals.

Recently, the security of navigation signals has been investigated and it has been shown that (civilian) Global Positioning System does suffer from a number of vulnerabilities, already exploited by adversaries for financial gain [36]. Most problems in these systems resulted from the lack of GPS signal authentication, and from receiver/signal design that allowed for the signals to be overshadowed by stronger (fake) signals. In response to this problem, Kuhn proposed a scheme [16] that relies on hidden spread spectrum coding and late key disclosure [23]. This scheme hides the GPS signal from the adversary using a secret key, which is fully (and publicly) disclosed at a later stage. This key is authenticated with the system public key, known to all receivers. That system prevents simple message forgery as GPS signals are now authenticated; however, replay of aggregated navigation signals with overshadowing remains a security problem for this system. Kuhn’s scheme, like all other schemes based on public-key infrastructures, requires appropriate set-up and maintenance of certificates and keys; this is not a major constraint of this

scheme, but still does limit its applicability in some scenarios (e.g., emergency and rescue) where the navigational infrastructure is set-up in an ad hoc manner.

To overcome message overshadowing, we propose the use of integrity codes for navigation signal authentication. We do realize that integrity coding is more appropriate for use with terrestrial navigation infrastructures than with GPS, because the terrestrial stations can provide stronger signal levels and more reliable coverage than satellites. We therefore consider scenarios in which terrestrial stations provide GPS-like navigational signals to mobile users (i.e. form Real-time Localization Systems (RTLS)). In such a scenario, each ground station provides the mobile node with its (i.e. station's) current location and signal sending time. A mobile node can be coarse-grain synchronized using the signal from a single station (in this case the propagation time is neglected). Since the signal (and the information about the current time in it) are encoded using integrity codes, signal overshadowing, as well as false signal insertion (or delay) will be detected at the receiver. Integrity coding therefore directly prevents attacks on time synchronization, provided that the receiver is aware of the presence of the navigation infrastructure and of the period in which the infrastructure is on (e.g., the navigation infrastructure can be permanently on).

In the case of localization signals, the mobile node measures the signal reception time and based on signal sending and reception times (i.e., the propagation time), estimates its distance to the base station. Based on three (or four) distance measurements, the node estimates its location in 2D (3D). Here, the same correction mechanisms as used in GPS [14] can be applied to avoid problems due to the lack of fine-grained synchronization between the infrastructure and the mobile node. In terms of robustness of this mechanism to attacks, since the infrastructure is permanently transmitting and the signals are integrity encoded, signal delays, overshadowing and fake signal insertion attacks will be detected at the receiver.

## VII. SECURITY ANALYSIS OF $I$ -CODES

In this section, we discuss the security of  $I$ -codes from the signal cancellation point of view. As we already mentioned in Section III, the security of  $I$ -codes depends on the inability of the attacker to flip symbols "1" into "0", by which she breaks the integrity of the exchanged messages. By a successful attack on  $I$ -codes, we consider that the attacker is able to break the integrity of the transmitted message, meaning that the receiver accepts a message as valid even if it has been modified by the attacker on the channel. Note that we reason about the security of  $I$ -codes within the system and the attacker model described in Section II.

We focus on the security of  $I$ -codes used over the radio communications channel. In order to delete (cancel) a signal  $s(t)$  emitted on a radio channel, the only hope for the adversary is to have her signal  $s'(t)$  arrive at the receiver with the same amplitude as  $s(t)$  but opposite in phase, that is,  $s'(t) = -s(t)$ . There are two main factors that make it hard for the attacker to cancel the signal at the receiver: (1) the unpredictability of the channel conditions (2) the unpredictability of the signal generated by the sender. In order to cancel the signal at the receiver, the attacker needs to estimate the channel conditions (to know how the channel will shape the original signal), and predict the shape of the signal generated at the sender (to know which form to generate to

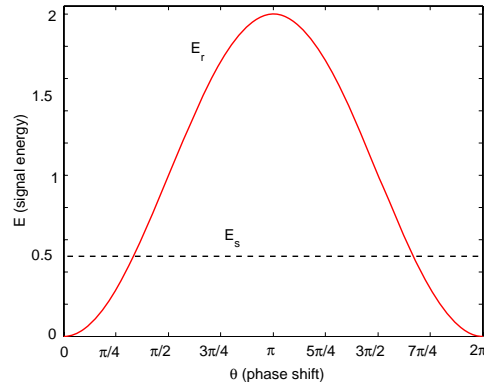


Fig. 11. Signal energy. The energy of the signal  $r(t) \equiv \cos(\omega_0 t) - \cos(\omega_0 t - \theta)$  and the signal  $s(t) = \cos(\omega_0 t)$  normalized with respect to  $T_s$  (the average power).

cancel the signal). Channel conditions are highly influenced by the environment and in high-frequency communication systems (e.g., 2.4 GHz), it is nearly impossible for the attacker to predict them due to the un-predictable amplitudes and phases, the multipath fading effects, etc.

In this section, we analyze how channel and signal unpredictability affect the attacker's ability to cancel-out the signal on the channel. We show that the odds of the adversary to cancel the signal  $s(t)$  are indeed negligible.

### A. Anti-Blocking Property of the Radio Channel

We first start by showing how channel conditions affect the attacker's ability to cancel the radio signal.

Let us assume that the sender emits cosine signal  $s(t)$  with unit amplitude and frequency  $f_0$ , i.e.,  $s(t) = \cos(\omega_0 t)$ , where  $\omega_0 = 2\pi f_0$ . We assume that the adversary knows somehow the exact value of the amplitude of the signal received at the receiver. Furthermore, we assume that there are no multipath fading effects and that the adversary knows  $s(t)$ . Note that with these assumptions, we only make the task of the adversary a lot easier. In reality, multipath effects and interferences from other transmitters can easily make the channel sufficiently random to forbid the attacker to even estimate the state of the signal at the receiver  $r(t)$ .

Let us define  $r(t) \equiv \cos(\omega_0 t) - \cos(\omega_0 t - \theta)$ , where  $\theta \in [0, 2\pi)$ . Here,  $r(t)$  can be thought of as the signal obtained as the superposition of the adversary's annihilating signal  $s'(t) = -\cos(\omega_0 t - \theta)$  and  $s(t)$ ;  $\theta$  accounts for the potential *phase shift*. The energy  $E_r$  of the signal  $r(t)$ , with duration  $T_s$ , can be calculated as follows [25]:

$$\begin{aligned} E_r &= \int_0^{T_s} r^2(t) dt \\ &= \frac{1}{\omega_0} \sin^2\left(\frac{\theta}{2}\right) (2\omega_0 T_s - \sin(\theta) + \sin(\theta - 2\omega_0 T_s)) \quad (6) \\ &\stackrel{(1)}{\approx} 2T_s \sin^2\left(\frac{\theta}{2}\right), \end{aligned}$$

where the approximation (1) is valid for high frequencies  $f_0$  (e.g.,  $f_0 = 2.4$  GHz), since  $-1 \leq \sin(\cdot) \leq 1$  implies  $\sin(\cdot)/\omega_0 = \sin(\cdot)/(2\pi f_0) \rightarrow 0$ .

We plot expression (6) in Figure 11; note that we normalize the energy with respect to  $T_s$  (therefore obtaining the average power of the signal). On the same figure, we also plot the

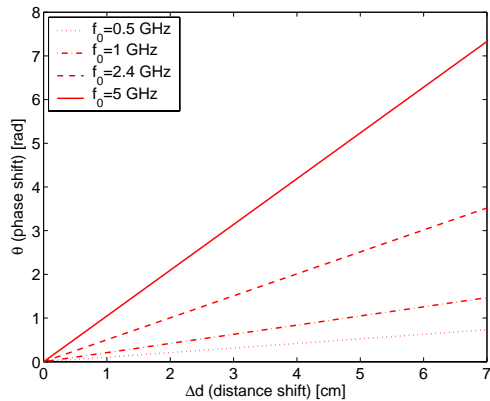


Fig. 12. Phase shift. The phase shift  $\theta$  as a function of the distance shift  $\Delta d$  for different frequencies  $f_0$ .

energy of the unobstructed signal  $s(t) = \cos(\omega_0 t)$ , i.e.,  $E_s = \int_0^{T_s} \cos^2(\omega_0 t) dt = T_s/2$ . A striking result on this figure is that for most values of  $\theta_0$  the adversary actually contributes to the energy of the original signal  $s(t)$ . In order to at least attenuate  $s(t)$ , the adversary has to ensure that  $\theta \in (-\theta_0, \theta_0)$ , where  $\theta_0$  is calculated as follows:

$$\frac{E_r}{E_s} = 4 \sin^2\left(\frac{\theta}{2}\right) < 1 \Rightarrow \sin\left(\frac{\theta}{2}\right) < \pm \frac{1}{2}, \quad (7)$$

and therefore,  $\theta_0 = 2 \arcsin(\frac{1}{2}) = \frac{\pi}{3}$ . Therefore, the attacker attenuates<sup>3</sup>  $s(t)$  for  $\theta \in [0, \frac{\pi}{3}] \cup (\frac{5\pi}{3}, 2\pi]$  (see Figure 11); note that this interval represents 1/3 ( $\approx 33\%$ ) of all the possible phase shifts.

We now show how demanding it is for the attacker to keep the phase shift  $\theta$  within the given bounds. We know that  $\theta = \omega_0 \Delta t$ , for a time shift (delay)  $\Delta t$ . In time  $\Delta t$ , the electromagnetic wave can travel the distance  $\Delta d = \Delta t \cdot c$ , where  $c$  is the propagation speed of the wave. We call  $\Delta d$  the *distance shift*. Combining these expressions we have:

$$\theta = \frac{2\pi f_0}{c} \Delta d. \quad (8)$$

In Figure 12 we plot expression (8) for different frequencies  $f_0$ . We can see that the higher the frequency of the signal is, the higher the impact of the fixed distance shift  $\Delta d$  on the phase shift  $\theta$  is. More importantly, for  $f_0 = 5$  GHz (IEEE 802.11a), a  $\Delta d$  as small as 1 cm results in phase shift of  $\frac{\pi}{3}$ . As we discussed above, the adversary has to ensure that  $\theta \in [0, \frac{\pi}{3}] \cup (\frac{5\pi}{3}, 2\pi]$ , in order to at least attenuate the signal  $s(t)$ . A more reasonable goal for the adversary would be to reduce the energy of the signal  $s(t)$  for say 50%, which requires, for  $f_0 = 5$  GHz,  $\theta \in [0, 0.7227] \cup (5.5605, 2\pi]$ . This phase shift corresponds to  $\Delta d \approx 7$  mm. Therefore, for high frequencies, the adversary has to estimate the distances between himself and both the sender and the receiver with a very high accuracy. Otherwise, she cannot hope to have the phase shift fall within the desired interval.

If the distance between the sender and the receiver continuously changes (in a fashion unpredictable to the attacker), the uncertainty of the adversary is further increased (note that this can be a very limited motion, in the order of  $\Delta d$ ). Therefore, in a sense, mobility helps security. Another source of the uncertainty for the adversary is the time delay  $\Delta t = \Delta d/c$ . For example, a distance shift  $\Delta d = 7$  mm is equivalent to a delay of  $\Delta t \approx 23$  ps.

<sup>3</sup>Not necessarily causing sufficient signal attenuation.

Therefore, the adversary has to operate with an extremely high time accuracy, otherwise he cannot keep  $\theta$  within the desired bounds.

Finally, if we assume that the receiver is equipped with two (or more) mutually separated antennas (as in *multiple antenna systems* [25]), then a signal from some transmitter will most likely arrive at the antennas with different phases. Moreover, this shift between the phases of the received signals will depend on the distances between the antennas as well as the relative position of the attacker with respect to the antennas. As we already saw above, at very high frequencies, even a very small distance shift will cause a significant phase shift. Any uncertainty in the distance shift (e.g., uncertainty regarding the positions of the antennas, etc.) implies uncertainty in the phase shift. We therefore conclude that it is reasonable to model phase shift  $\theta$  by a random variable with appropriate distribution.

### B. Randomization at the Sender

We already saw in Figure 11 that for 1/3 of the possible phase shifts, the adversary actually attenuates the sender's signal. Therefore, when using only a single waveform (e.g.,  $\cos(\omega_0 t)$ ) during the whole period  $T_s$ , the adversary may have a non-negligible probability to attenuate the desired signal. For example, assuming  $\theta$  is a sample of a random variable  $\Theta$  with uniform distribution on  $[0, 2\pi)$ , the adversary attenuates the signal in the single time interval  $T_s$  with probability 1/3. We now show how to make this probability satisfactorily small. The idea is to split the symbol interval  $T_s$  into  $K$  smaller and equal time slots  $T_m$  when the symbol "1" is to be sent. Then, for each *mini-slot*  $T_m$ , the sender generates a signal with the phase chosen uniformly at random from  $[0, 2\pi)$  and emits these  $K$  signals on the channel during the time  $T_s$ . For example, these  $K$  signals can be described by the following random process  $S(t) = \cos(\omega_0 t + \Phi)$ , where  $\Phi$  is a random variable with uniform distribution on  $[0, 2\pi)$ .

From the discussion in the previous section, it is reasonable to model the phase shift as a random variable  $\Theta$ . Let us assume  $\Theta$  to be uniformly distributed on  $[0, 2\pi)$ . Let  $p_\alpha$  be the probability that the adversary attenuates the signal emitted in a given mini-time slot for at least  $(1 - \alpha) \times 100\%$ , that is,  $E_r/E_s \leq \alpha$ , where  $\alpha \in [0, 1]$ . We say that any such mini-slot signal is  $\alpha$ -attenuated<sup>4</sup>. For  $\Theta$  uniform random variable, i.e.  $f_\Theta(\theta) = \frac{1}{2\pi}$ , we have

$$\begin{aligned} p_\alpha &= Pr \left[ \frac{E_r}{E_s} \leq \alpha \right] \\ &\stackrel{(1)}{=} Pr \left[ \sin\left(\frac{\theta}{2}\right) \leq \pm \frac{\sqrt{\alpha}}{2} \right] \\ &= Pr [\theta \in [0, \theta_\alpha] \cup (2\pi - \theta_\alpha, 2\pi)] \\ &\stackrel{(2)}{=} \frac{\theta_\alpha}{\pi}, \end{aligned} \quad (9)$$

where  $\theta_\alpha = 2 \arcsin(\sqrt{\alpha}/2)$ , the equality (1) follows from expression (7), and the equality (2) follows from the distribution of  $\Theta$ .

We further note that  $\Phi$  and  $\Theta$  are independent random variables; indeed,  $\Theta$  models the inability of the adversary to perfectly estimate the required distances and/or any delay that the adversary introduces. Therefore,  $p_\alpha$  (as given in expression (9)), is the same

<sup>4</sup>Note that even if the adversary does attenuate the energy of the original signal  $s(t)$  by 50%, the average power as measured by the receiver may still be well above the threshold  $P_0$ .

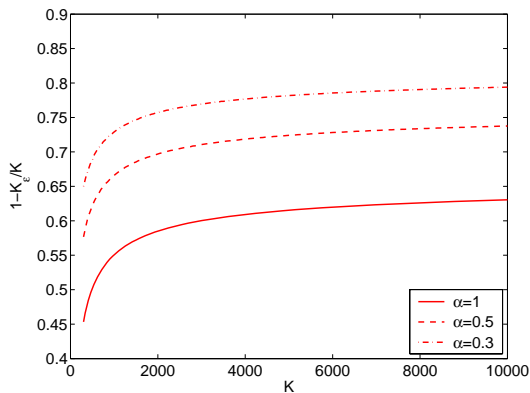


Fig. 13. Attenuated mini-slot signals. The ratio of mini-slot signals that are not  $\alpha$ -attenuated as a function of  $K$ ;  $\epsilon = 10^{-14}$ .

for all the  $K$  mini-slots. Then, for the fixed time interval  $T_s$ , the probability that the number  $K_a$  of  $\alpha$ -attenuated mini-slot signals is exactly  $k \leq K$ , can be calculated from the binomial distribution with parameters  $p = p_\alpha$  and  $q = 1 - p_\alpha$  as follows

$$Pr[K_a = k] = \binom{K}{k} \frac{1}{\pi^K} \theta_\alpha^k (\pi - \theta_\alpha)^{K-k}, \quad (10)$$

where  $\theta_\alpha = 2 \arcsin(\sqrt{\alpha}/2)$ . For the binomial distribution (10), we can calculate the expected ratio  $K_a/K$  of the  $\alpha$ -attenuated mini-slots as follows,

$$E\left[\frac{K_a}{K}\right] = \frac{E[K_a]}{K} = \frac{\theta_\alpha}{\pi} \leq \frac{1}{3}, \quad (11)$$

where the last inequality follows from the fact that  $\theta_\alpha \leq \theta_1 = \frac{\pi}{3}$ . Therefore, on average, at most 1/3 of the total number of mini-slot signals will be  $\alpha$ -attenuated, i.e.,  $E_r/E_s \leq \alpha$ .

Note, however, that the expected value of the ratio  $K_a/K$  is independent of  $K$ , and therefore it does not give any useful information about the role of  $K$  and what value we should choose for it. We next study this aspect. Let us denote with  $K_\epsilon$  ( $K_\epsilon \leq K$ ) the *smallest* threshold for which the following holds

$$Pr[K_a \leq K_\epsilon] \geq 1 - \epsilon, \quad (12)$$

where  $\epsilon \in [0, 1]$ . Note that  $Pr[K_a \leq K_\epsilon] = \sum_{k=0}^{K_\epsilon} Pr[K_a = k]$ , with  $Pr[K_a = k]$  given by (10). Note further that  $Pr[K_a \leq K_\epsilon]$  is related to a single time interval  $T_s$  during which the symbol “1” is transmitted. By the independence, the probability  $Pr^n[K_a \leq K_\epsilon]$  that  $K_a \leq K_\epsilon$  after  $n$  symbol “1” transmissions ( $n$  time intervals  $T_s$ ) satisfies

$$Pr^n[K_a \leq K_\epsilon] \geq (1 - \epsilon)^n \approx e^{-n\epsilon},$$

where the last approximation is valid for small  $\epsilon$ . For the given  $n$ , by choosing  $\epsilon$  such that  $e^{-n\epsilon}$  is reasonably close to 1, we essentially make  $K_\epsilon$  an “upper bound” on the number of mini-slot signals that are  $\alpha$ -attenuated in any given time slot  $T_s$  (out of the total of  $n$  slots). Likewise,  $(K - K_\epsilon)$  provides a “lower bound” on the number of mini-slot signals that are not  $\alpha$ -attenuated.

In Figure 13, we plot the ratio  $(1 - K_\epsilon/K)$  of the mini-slot signals that are not  $\alpha$ -attenuated as a function of  $K$ , for  $\epsilon = 10^{-14}$ . For  $n = 10^{10}$ , we have  $e^{-n\epsilon} \approx 0.9999$ , i.e., even after as many as  $10^{10}$  transmissions of the symbol “1”, the probability that  $K_a \leq K_\epsilon$  is at least 0.9999. If we transmit on average one symbol “1” per second (meaning that we do nothing else but transmitting such signals), then it takes around 310 years to see

all the  $n$  symbols. In this case, the smallest  $K_\epsilon$  for which the bound (12) holds, is a reasonable upper bound on  $K_a$ . Coming back to Figure 13, we can see that if  $K$  is set too low, we cannot achieve a very high ratio of non  $\alpha$ -attenuated mini-slot signals for all  $n$  transmissions of the symbol “1”. Therefore,  $K$  should be chosen based on the expected  $\alpha$  and the desired ratio  $1 - K_\epsilon/K$ .

### C. Energy Content of the Emitted Signals

We already argued that it is reasonable to model the phase shift as a random variable  $\Theta \in [0, 2\pi)$ . It is then interesting to calculate the energy of the resulting random signal. Let us define a random process  $R(t) = \cos(\omega_0 t) - \cos(\omega_0 t - \Theta)$ . We will calculate the energy of this process when  $\Theta$  has uniform distribution on  $[0, 2\pi)$ . We have  $f_\Theta(\theta) = \frac{1}{2\pi}$ ,  $\forall \theta \in [0, 2\pi)$ . The energy content  $\mathcal{E}_R$  of the random process  $R(t)$ , in the time interval  $T$ , is defined as [25]:

$$\mathcal{E}_R = E\left[\int_0^T R^2(t) dt\right] = \int_0^T E[R^2(t)] dt. \quad (13)$$

Now, for  $E[R^2(t)]$  we have:

$$\begin{aligned} E[R^2(t)] &= \int_0^{2\pi} r^2(t) f_\Theta(\theta) d\theta \\ &= \frac{1}{2\pi} \int_0^{2\pi} (\cos(\omega_0 t) - \cos(\omega_0 t - \theta))^2 d\theta \\ &= 1 + \frac{1}{2} \cos(2\omega_0 t). \end{aligned} \quad (14)$$

Plugging this into the expression (13), we obtain:

$$\mathcal{E}_R = T + \frac{\sin(2\omega_0 T)}{4\omega_0} \stackrel{(1)}{\approx} T, \quad (15)$$

where (1) is valid for high frequencies  $f_0$ , since  $-1 \leq \sin(\cdot) \leq 1$  implies  $\sin(\cdot)/(4\omega_0) = \sin(\cdot)/(8\pi f_0) \rightarrow 0$ . Therefore, on average, the adversary only increases the energy of the resulting signal  $r(t)$ ; the energy content of  $r(t)$  without the adversary is  $T/2$  (Figure 11).

From the analysis in this section, we conclude that we can easily ensure that the adversary cannot block the symbol “1” emitted over a radio channel, even under very advantageous assumptions for him (i.e., no multipath fading effects, perfect estimate of signal amplitudes, etc.).

## VIII. RELATED WORK

Here, we review work in the context of secure communication over insecure (wireless) channels. In [29] and [28], Stajano and Anderson propose the *resurrecting duckling* security policy model, in which key establishment is based on the physical contact between communicating parties (their PDAs). The potential drawback of this approach is that the realization of physical contact can be cumbersome with bulky devices (e.g., laptops). An approach inspired by the resurrecting duckling security policy model is proposed by Balfanz et al. [6]. In this work, the authors relax the requirement that the location limited channel has to be secure against passive eavesdropping; they introduce the notion of a *location-limited channel*. A location-limited channel is used to exchange pre-authentication data and should be resistant to active attacks (e.g., man-in-the-middle). Possible candidates for a location-limited channel include: physical contact, infrared, and ultrasound [6]. Here again, the disadvantage of this approach is that it may be cumbersome to realize a link with bulky

devices (e.g., laptops) in the case of infrared or physical contact. Our key establishment mechanisms based on *I*-codes enable key establishment over a radio channel in a more practical way for the user as no physical contact is required. Asokan and Ginzboorg propose another solution based on a shared password [5]. They consider the problem of setting up a session key between a group of people (i.e., their computers) who get together in a meeting room and who share no prior context except a fresh password.

In [24], Perrig and Song suggest using hash visualization to improve the security of such systems. Hash visualization is a technique that replaces meaningless strings with structured images. However, having to compare complex images can be cumbersome. In US patent no. 5,450,493 [20], Maher presents several methods to verify DH public parameters exchanged between users. This technique had a flaw, discovered by Jakobsson [17]. Motivated by the flaw, Larsson and Jakobsson [17] proposed two solutions based on a temporary secret shared between the two users. In [12] and [13], Gehrman et al., propose a set of techniques to enable wireless devices to authenticate one another via an insecure radio channel with the aid of the manual transfer of data between the devices. In [32], Čagalj et al. propose an optimal message authenticator, a more efficient protocol that enables provably secure authentication through the transfer of a short bit sequence over an authenticated channel, and a set of techniques for key establishment over a radio link in peer-to-peer networks based on the Diffie-Hellman protocol. In [4], Alpern and Schneider present a protocol that allows two parties to agree on a secret key on channels for which an adversary cannot tell who is the source of each message. As a follow-up, in [10], Castelluccia and Mutaf propose two movement-based pairing protocols for CPU-constrained devices. We should mention the work of Corner and Noble [11], who consider the problem of transient authentication between a user and his device, as well as the work of Čapkun et al. [35], where the authors show how to make use of users mobility to bootstrap secure communication in ad hoc networks. We also acknowledge the contribution of Perrig et al. in [23], where the authors propose Tesla, a protocol for broadcast authentication based on delayed key disclosure, and the contribution of Maurer [22], who studied secret key agreement by public discussion, based on common information.

## IX. CONCLUSION

In this work, we introduced *integrity (I) codes*, a novel mechanism that enables integrity protection of messages exchanged over a radio channel between entities that do not hold any mutual authentication material (i.e. public keys or shared secret keys). We have analyzed *I*-codes in detail and we have shown that they are secure in a realistic attacker model. We further introduced a novel mechanism, called *authentication through presence* based on *I*-codes. We demonstrated the use of this mechanism in a number of application scenarios including broadcast authentication, key establishment and secure navigation. We implemented *I*-codes on several wireless communication platforms; we showed that *I*-codes can be implemented efficiently and at a low cost with widely available platforms and hardware components.

## REFERENCES

- [1] Gnu radio – the gnu software radio. <http://www.gnu.org/software/gnuradio/index.html>.
- [2] Madwifi: Multiband atheros driver for wireless fidelity. <http://madwifi.org/>.
- [3] Mica sensor platform. <http://www.xbow.com>.
- [4] B. Alpern and F. Schneider. Key exchange using Keyless Cryptography. *Information processing letters*, 16(2):79–82, 1983.
- [5] N. Asokan and P. Ginzboorg. Key Agreement in Ad-hoc Networks. *Computer Communications*, 23(17):1627–1637, November 2000.
- [6] D. Balfanz, D. Smetters, P. Stewart, and H. Wong. Talking to Strangers: Authentication in Ad-Hoc Wireless Networks. In *Proceedings of the 9th Annual Network and Distributed System Security Symposium (NDSS)*, 2002.
- [7] J. M. Berger. A Note on Error Detecting Codes for Asymmetric Channel. *Information and Control*, 4:68–73, 1961.
- [8] M. Blaum and H. van Tilborg. On *t*-Error Correcting/All Unidirectional Error Detecting Codes. *IEEE Transactions on Computers*, pages 1493–1501, 1989.
- [9] J. M. Borden. Optimal Asymmetric Error Detecting Codes. *Information and Control*, 53:66–73, 1982.
- [10] C. Castelluccia and P. Mutaf. Shake Them Up! A movement-based pairing protocol for CPU-constrained devices. In *Proceedings of the ACM Conference on Mobile Systems, Applications and Services (MobiSys)*, 2005.
- [11] M. Corner and B. Noble. Protecting applications with transient authentication. In *First ACM/USENIX International Conference on Mobile Systems, Applications and Services (MobiSys'03)*, San Francisco, CA, May 2003.
- [12] C. Gehrman, C.J. Mitchell, and K. Nyberg. Manual Authentication for Wireless Devices, January 2004. RSA Cryptobytes, Vol. 7, No. 1.
- [13] C. Gehrman and K. Nyberg. Enhancements to Bluetooth Baseband Security. In *Proceedings of Nordsec*, Copenhagen, Denmark, November 2001.
- [14] I. Getting. The Global Positioning System. *IEEE Spectrum*, December 1993.
- [15] S. Han, R. Rengaswamy, R. Shea, E. Kohler, and M. Srivastava. A Dynamic Operating System for Sensor Nodes. In *Proceedings of the ACM Conference on Mobile Systems, Applications and Services (MobiSys)*, 2005.
- [16] M. G. Kuhn. An Asymmetric Security Mechanism for Navigation Signals. In *Proceedings of the Information Hiding Workshop*, 2004.
- [17] J.-O. Larsson and M. Jakobsson. SHAKE. Private communication with M. Jakobsson.
- [18] E. L. Leiss. Data Integrity on Digital Optical Discs. *IEEE Transactions on Computers*, 33:818–827, 1984.
- [19] Arjen K. Lenstra and Eric R. Verheul. Selecting Cryptographic Key Sizes. *Journal of Cryptology*, 14(4):255–293, 2001.
- [20] D.P. Maher. United States Patent (No. 5,450,493): Secure communication method and apparatus. <http://www.uspto.gov>, 1993.
- [21] Wenbo Mao. *Modern Cryptography, Theory & Practice*. Prentice Hall PTR, 2004.
- [22] Ueli Maurer. Protocols for secret key agreement by public discussion based on common information. In *Advances in Cryptology — CRYPTO '92*, volume 740 of *Lecture Notes in Computer Science*, pages 461–470. Springer-Verlag, August 1993.
- [23] A. Perrig, R. Canetti, J.D. Tygar, and D. Song. The TESLA Broadcast Authentication Protocol. *RSA CryptoBytes*, 5(Summer), 2002.
- [24] A. Perrig and D. Song. Hash Visualization: A New Technique to Improve Real-World Security. In *Proceedings of the 1999 International Workshop on Cryptographic Techniques and E-Commerce (CrypTEC '99)*, pages 131–138, July 1999.
- [25] John G. Proakis and Masoud Salehi. *Communication Systems Engineering – Second Edition*. Prentice Hall, 2002.
- [26] K. Rasmussen, S. Čapkun, and M. Čagalj. SecNav: Secure Broadcast Localization and Time Synchronization in Wireless Networks (extended abstract). In *Proceedings of the ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, 2007.
- [27] J. M. Berger. A note on an error detection code for asymmetric channels. *Information and Control*, 4, March 1961.
- [28] F. Stajano and R. Anderson. The Resurrecting Duckling: Security Issues for Ad-hoc Wireless Networks. In *Proceedings of the 7th International Workshop on Security Protocols*, 1999.
- [29] Frank Stajano. *Security for Ubiquitous Computing*. John Wiley & Sons, Ltd., 2002.
- [30] Symantec. Securing enterprise wireless networks. White Paper, 2003.
- [31] D. L. Tao, R. P. Hartmann, and P. K. Lala. An efficient class of unidirectional error detecting/correcting codes. *IEEE Transactions on Computers*, 37(7):879–882, 1988.

- [32] M. Čagalj, S. Čapkun, and J.-P. Hubaux. Key Agreement in Peer-to-Peer Wireless Networks. *Proceedings of the IEEE (Special Issue on Cryptography and Security)*, 94(2), 2006.
- [33] M. Čagalj, S. Čapkun, RamKumar Rengaswamy, Ilias Tsigkogiannis, M. Srivastava, and Jean-Pierre Hubaux. Integrity (I) codes: Message Integrity Protection and Authentication Over Insecure Channels. Oakland, California, USA, 2006.
- [34] S. Čapkun and J.-P. Hubaux. Secure Positioning in Wireless Networks. *IEEE Journal on Selected Areas in Communications*, 24(2), February 2006.
- [35] S. Čapkun, J.-P. Hubaux, and L. Buttyán. Mobility Helps Peer-to-Peer Security. *IEEE Transactions on Mobile Computing*, 2006.
- [36] J. S. Warner and R. G. Johnston. Think GPS Cargo Tracking = High Security? Think Again. *Technical report, Los Alamos National Laboratory*, 2003.



**Ilias Tsigkogiannis** received his B.Sc. degree in Electrical and Computer Engineering from the University of Patras, Greece in 2003. From October 2003 to June 2004 he worked as a researcher in the Applied Electronics Laboratory in the University of Patras. In June 2006 he received his M.Sc. degree in Electrical Engineering from University of California, Los Angeles. From June 2005 to June 2006 he worked as an intern software developer for Symantec Corporation in Los Angeles. Since July 2006 he's been working as a software development engineer for Microsoft Corporation in Redmond, WA. His research interests are in networked embedded systems and sensor networks.



**Srdjan Čapkun** received the Dipl Ing degree in electrical engineering and computer science from the University of Split, Croatia, in 1998 and the PhD degree in communication systems from the Swiss Federal Institute of Technology in Lausanne (EPFL) in 2004. He is currently an assistant professor in the Department of Computer Science, ETH Zurich. Prior to joining ETH Zurich, he was a postdoctoral researcher in the Networked and Embedded Systems Laboratory (NESL), University of California, Los Angeles, and an assistant professor in the Informatics and Mathematical Modeling Department, Technical University of Denmark (IMM DTU). His research interests include the design and analysis of security protocols for wireless and wireline networks.



**Jean-Pierre Hubaux** joined the faculty of EPFL in 1990. His research activity is focused on wireless networks, with a special interest in security and cooperation issues.

In 1991, he designed the first curriculum in communication systems at EPFL. He was promoted to full professor in 1996. In 1999, he defined some of the main ideas of the National Competence Center in Research named "Mobile Information and Communication Systems" (NCCR/MICS); this center (still very active nowadays) is often nicknamed "the Terminodes project". In this framework, he has notably defined, in close collaboration with his students, novel schemes for the security and cooperation in wireless networks; in particular, he has devised new techniques for key management, secure positioning, and incentives for cooperation in such networks. In 2003, he identified the security of vehicular networks as one of the main research challenges for real-world mobile ad hoc networks. In 2007, he completed a graduate textbook entitled "Security and Cooperation in Wireless Networks" (Cambridge University Press), with Levente Buttyan.

He is the chairman of the steering committees of ACM WiSec and of ACM Mobihoc and a member of the steering committee of IEEE Transactions on Mobile Computing. He has been serving on the program committees of numerous conferences and workshops, including SIGCOMM, Infocom, Mobicom, Mobihoc, SenSys, WiSe, and VANET. He is a member of the Federal Communications Commission (ComCom), the "Swiss FCC".

He held visiting positions at the IBM T.J. Watson Research Center and at UC Berkeley.



**Mario Čagalj** received the Dipl.Ing. degree in Computer Science and Electrical Engineering from University of Split, Croatia, in 1998, and the Ph.D. degree in Communication Systems from the Ecole Polytechnique Fédérale de Lausanne (EPFL) in February 2006. In 2001 he finished the Pre-doctoral School in Communication Systems, EPFL. From 2001 to 2006 he was a Research Assistant in the Laboratory for computer Communications and Applications (LCA) at EPFL. In January 2006 he joined the Faculty of Electrical Engineering, Mechanical

Engineering and Naval Architecture (FESB) at the University of Split. He is currently an Assistant Professor in the Department of Electronics.

His research interests include the design and analysis of security protocols for wireless and wireline networks, application of game theory to communication networks and the design of energy-efficient communication protocols for wireless networks. For more details please check <http://www.fesb.hr/~mcagalj>.



**Mani Srivastava** is on the faculty at UCLA where he is Professor and Vice Chair of Electrical Engineering, and Professor of Computer Science. He is also associated with UCLA's Center for Embedded Networked Sensing (CENS) where he co-leads the Systems Area Research. Previously, he did his undergraduate studies at the Indian Institute of Technology, Kanpur, and his M.S. and Ph.D. at the University of California, Berkeley, and worked at ATT/Lucent Bell Labs Research from 1992-1996.

His research interests are in power-aware computing and communications, wireless and mobile systems, embedded computing, pervasive sensing, and their applications in urban, social, and personal contexts. He has served as the Editor-in-Chief of the ACM Sigmobile Mobile Computing and Communications Review, and as Associate Editor for IEEE Transactions on Mobile Computing, IEEE/ACM Transactions on Networking, and the ACM Transactions on Sensor Networks. He has served as the TPC Co-chair for ACM MobiHoc, ACM SenSys, and IEEE/ACM IPSN. He has received the President of India's Gold Medal for academic excellence, the NSF CAREER Award, the Okawa Foundation Grant, Best Paper Award at the IEEE ICDCS, and Design Contest and Demonstration Awards at IEEE/ACM DAC, IEEE ISSCC, IEEE/ACM ISLPED, and IEEE/ACM IPSN.



**Ram Kumar** graduated with a PhD in Electrical Engineering from UCLA in Spring 2007. His broad research interests are in design of software systems for embedded platforms. His thesis was on memory protection techniques for sensor network applications. He is one of the core developers of the SOS operating system that received the best demo award at IPSN 2005. He received his MS from UCLA in 2003 and his B.Tech from Indian Institute of Technology, Delhi in 2001. He is currently working on distributed software systems at Google in Mountain