Integrity Regions: Authentication Through Presence in Wireless Networks

Srdjan Čapkun, *Member, IEEE,* Mario Čagalj, *Member, IEEE,* Ghassan Karame, *Member, IEEE,* Nils Ole Tippenhauer, *Member, IEEE*

Abstract—Despite years of intensive research, the main deterrents of widely deploying secure communication between wireless nodes remains the cumbersome key setup process. In this paper, we address this problem and we introduce *Integrity (I) regions*, a novel security primitive that enables message authentication in wireless networks without the use of pre-established or pre-certified keys. Integrity regions are based on the verification of entity proximity through time-of-arrival ranging techniques. IRegions can be efficiently implemented with ultrasonic ranging, in spite of the fact that ultrasound ranging techniques are vulnerable to distance enlargement and reduction attacks. We further show how IRegions can be used for key establishment in mobile peer-to-peer wireless networks and we propose a novel automatic key establishment approach, largely transparent to users, by leveraging on IRegions and nodes' mobility. We analyze our proposals against a multitude of security threats and we validate our findings via extensive simulations.

Index Terms—Authentication, Key Establishment, Distance Bounding, Node Mobility, Wireless Networks.

1 Introduction

Wireless technology enables users to communicate, regardless of their micro-location, provided that their devices are located within each others' radio range. The extent of this range depends on devices' transmitting power, (transmitting and receiving) antenna gains, receiver sensitivity, and on various environmental factors [1]. In an adversarial setting, this means that users can hardly predict the maximal distance from which an adversary with a much higher receiver sensitivity and antenna gain can eavesdrop on their communication. Moreover, an adversary can use devices with high transmission power; this enables her to insert, modify, and jam messages exchanged between users from large distances. Given this, two wireless devices trying to establish a shared secret key (e.g. using the Diffie-Hellman (DH) protocol [2]) could be subject to the known man-in-the-middle (MITM) attacks on the non-authenticated DH protocol [3] and therefore might not be able to establish the key securely.

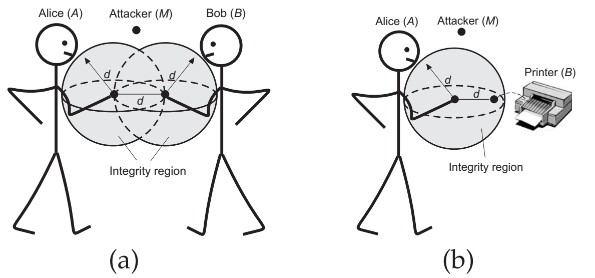
However, in spite of its aforementioned shortcomings, the legacy paradigm of wireless communication embeds basic properties that might *strengthen* secure communication between nodes; indeed, wireless networks can leverage on node mobility to *increase* security. People want to meet in person because they believe – and rightly so – that physical presence is the best way to increase trust with others and to exchange information in a reliable way. To achieve this, they might accept to travel long distances and to support the related inconveniences.

In this work, we leverage on this observation and we introduce *Integrity regions (IRegions)*, a new security primitive that prevents MITM attacks on wireless communication through *verification of presence*. IRegions rely on range measurements to prevent distant attackers from inserting forged messages into the communication between the devices. In our approach, for each received message, the devices verify if it came from an expected distance (i.e. from their integrity regions). If the message came from the expected (safe) distance, it is accepted as authentic. With IRegions, we force the attacker to be present in the close proximity of the devices in order to perform a man-in-the-middle attack. If the users can verify (i.e., visually) that the attacker is not present in their immediate vicinity, they will be able to verify the integrity and the authenticity of the exchanged messages and therefore prevent MITM attacks. IRegions can be implemented with ultrasonic or radio time-of-arrival ranging techniques. However, due to its lower hardware requirements, we demonstrate in this work the feasibility of IRegions using ultrasonic ranging.

The most straightforward application of IRegions lies in the area of key establishment in wireless peer-to-peer networks; using IRegions, users can establish a shared secret key by simply getting physically close (e.g., by meeting each other) and by verifying that the attacker is absent from their close proximity (the joint integrity region). Fig. 1 (a) illustrates the integrity regions of users A and B (whose sizes are determined by their mutual distance d), and the location of attacker (M) placed outside of these integrity regions. In this setting, users A and B can, using IRegions, securely establish their shared secret key. As illustrated in Fig. 1 (b), IRegions can also be used to acquire authentic public keys from other devices. Another application of IRegions

- *S. Čapkun, G. Karame and N.O. Tippenhauer are affiliated with the Department of Computer Science, ETH Zurich, 8092 Switzerland. E-mail: {capkuns,karameg,tinils}@inf.ethz.ch*
- *M. Čagalj is affiliated with the FESB, University of Split, 21000 Croatia. E-mail: mario.cagalj@fesb.hr*

Digital Object Indentifier 10.1109/TMC.2010.127



(a) (b)

Fig. 1. Examples of applications of integrity regions. (a) Key establishment (e.g., exchange of the Diffie-Hellman public keys – bidirectional message authentication and integrity verification); (b) Device authentication (user receives an authentic public-key of a device – unidirectional message authentication and integrity verification).

lies in the area of access control protocols for implantable medical devices (IMDs, e.g., pacemakers) [4]. IRegions can enable an implanted medical device to grant access to its resources only to those devices that are in its close proximity (current protocols require that the reader is located within few centimeters from the reader during the key exchange phase [4]).

Even in scenarios in which there is no visual verification of the space by users, integrity regions force the attacker to come close to the devices (i.e., within devices' mutual distance) in order to successfully perform MITM attacks. If the exchanged messages are part of a persistent key establishment system, the attacker has to be inside the integrity region of the nodes whenever they exchange messages. As a result, the cost of mounting the attack can exceed by far the expected payoff (gain) for the attacker, thus rendering the attack unattractive.

This paper is an enhanced and extended version of the work in [5] (we omitted some details due to lack of space). More specifically, in this paper we provide formal proofs for the security of our scheme and we extend the concept of IRegions to mobile wireless peer-to-peer networks. In that respect, we propose novel and automatic protocols that allow secure key establishment between users by leveraging on the mobility of nodes. We thoroughly analyze the security of our proposals and we evaluate their performance through extensive simulations using realistic mobility models.

The rest of the paper is organized as follows. In Section 2, we state our problem, overview related work and we describe our system and attacker model. In Section 3, we formally introduce IRegions. In Section 4, we show how IRegions can be used to securely establish keys between two users. In Section 5, we utilize the concept of IRegions in the scope of key establishment in wireless peer-to-peer networks. In Section 6, we discuss further insights to our proposed scheme and we conclude the paper in Section 7.

2 Problem statement

We consider examples of people meeting and gathering in public places (e.g., shopping malls, restaurants, conferences, classrooms, etc.) and continue communicating after their encounters (typically over the Internet).

Now consider the following problem. Two users meet for the first time and want to establish a shared key (alternatively, a user approaches a device and wants to establish a key with this device). Although they can visually recognize each other, we assume that they do not share any authenticated cryptographic information (e.g., public keys or a shared secret) prior to this meeting. In addition, the users can communicate only over a radio channel (no infrared or physical ports are available).

Note that an attacker does not have to be located close to the devices running the protocol. This is due to two reasons: *(i)* an attacker can use high-gain antennas to eavesdrop on devices' communication and *(ii)* an attacker can jam devices' communication from large distances, using high-power antennas. With these two "tools" in hand, an attacker's *attack space* is large and she effectively controls the channel [6], [7].

In this work, our goal is *to devise mechanisms that prevent the attacker from modifying the messages containing public keys and/or DH parameters on a radio channel without being noticed.*

2.1 Related work

The problem of key establishment is a very active area of research. However, thus far, only few solutions have been proposed for the problem that we consider.

Stajano *et al.* propose the *resurrecting duckling* security policy model, [8] and [9], in which key establishment is based on the physical contact between communicating parties (their PDAs). An approach inspired by the resurrecting duckling security policy model was also proposed by Balfanz *et al.* [10]. Asokan *et al.* propose another solution based on a shared password [11]. They consider the problem of setting up a session key between a group of people (i.e., their computers) based on a fresh password chosen and shared among the parties present in the room; the shared password is then used to derive a strong shared session key. Users might be unreliable when dealing with meaningless strings, and have difficulties remembering strong passwords. To counter this problem, in [12], Perrig *et al.* suggest using hash visualization to improve the security of such systems.

In [13], Maher *et al.* present several methods to verify DH public parameters exchanged between users. This technique had a flaw, discovered by Jakobsson. Motivated by the flaw, Jakobsson *et al.* [14] proposed two solutions based on a temporary secret shared between the two users. In [15], Čagalj *et al.* propose an efficient protocol that enables provably secure authentication through the transfer of a short bit sequence over specifically constructed secure channels. In [16], Laur *et al.* show that such message authenticators require non-malleable commitment schemes. In [17], Castelluccia *et al.* propose a device pairing protocol for CPU-constrained devices which relies on device indistinguishability. In [4], Rasmussen *et al.* proposed an access control protocol for implantable medical devices (IMDs).

In [18], Čagalj *et al.* propose Integrity codes, a coding scheme that enables integrity protection of messages exchanged between entities that do not hold any mutual authentication material (i.e. public keys or shared secret keys). In [19] McCune, Perrig and Reiter present a system that utilizes barcodes and camera-telephones to implement a visual channel for authentication and demonstrative identification of devices. In [20], Goodrich *et al.* present a similar solution, based on voice channels.

In the scope of mobile ad-hoc networks, a number of proposals have been recently published to bootstrap the keys [21], [22], [23], [24], [25]. However, they diverge from our main goal, which is to set up a shared key for future use over the Internet. Other approaches, motivated by the mobility in IPv6, proposed by Montenegro and Castelluccia (SUCV) [26] and by O'Shea and Roe (CAM) [27], consist in binding the public key of a node to its IP address. However, all these solutions require an active involvement of the user (they are thus cumbersome to use), and thus they may not scale well.

2.2 Primitives

The Diffie-Hellman (DH) key agreement protocol [2] seems to be appropriate for the problem at hand; to agree on a shared key, two users, Alice (A) and Bob (B) proceed as follows: A picks a random secret exponent X_A, and calculates the DH public parameter g^{X_A}, where g is a generator of a group of large order. B does the same, that is, he calculates g^{X_B}. Finally, A and B exchange the public parameters g^{X_A} and g^{X_B} and calculate the shared DH key $K = g^{X_A X_B} = (g^{X_A})^{X_B} = (g^{X_B})^{X_A}$. In this sense, the DH key agreement protocol is believed to be secure against a passive eavesdropping adversary[1].

We present our solution over the multiplicative group $\mathbb{G}$ with the generator g. Here, we take $\mathbb{G}$ to be a subgroup of $\mathbb{Z}_p^*$ of the prime order q, where $\mathbb{Z}_p^*$ is the multiplicative group of non-zero integers modulo a large prime p. However, our analysis equally applies to any group in which the Diffie-Hellman problem is hard. These are all groups in which it is infeasible to distinguish between quadruples of the form (g, g^x, g^y, g^{xy}) and quadruples (g, g^x, g^y, g^z) where x, y, z are random exponents. Furthermore, we assume that p and a generator g of $\mathbb{Z}_p^*$, $(2 \leq g \leq p-2)$ are selected and published. We further assume that all devices are preloaded with these values[2].

Commitment schemes are additional important cryptographic building blocks that we will be using in our protocols. The semantics of a commitment scheme are the following: *(i)* a user who commits to a certain value cannot change this value afterwards (we say that the scheme is binding), *(ii)* the commitment is hidden from its receiver until the sender "opens" it (we say that the scheme is hiding). That is, a commitment scheme transforms a value m into a commitment/opening pair (c, o), where c reveals no information about m, but (c, o) together reveal m, and it is infeasible to find $\hat{o}$ such that $(c, \hat{o})$ reveals $\hat{m} \neq m$. Now, if Bob wants to commit a value m to Alice, he first generates the commitment/opening pair $(c, o) \leftarrow \mathrm{commit}(m)$, and sends c to Alice. To open m, Bob simply sends $\hat{o}$ (and m if necessary) to Alice, who runs $\hat{m} \leftarrow \mathrm{open}(\hat{c}, \hat{o})$; we denote with $\hat{m}$ the message at the receiver's side when message m is sent over a public (unauthentic) channel. If the employed commitment scheme is "correct", we must have $m = \hat{m}$ at the end of the protocol.

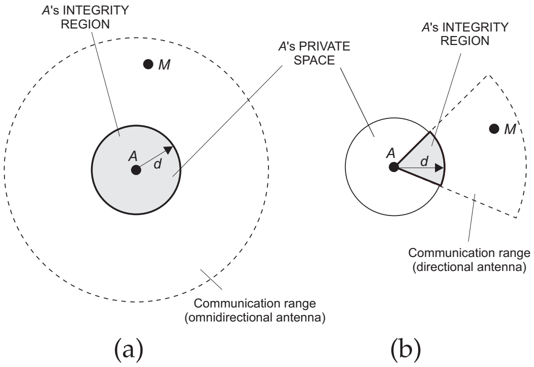

A's INTEGRITY REGION
M
A's PRIVATE SPACE
A
d
Communication range (omnidirectional antenna)
(a)
A's INTEGRITY REGION
M
A
d
Communication range (directional antenna)
(b)

Fig. 2. Examples of integrity regions for user A with (a) omnidirectional and (b) directional antennas.

2.3 System and Attacker Model

We assume that the two entities involved in the communication (A and B) know the (public) protocol parameters. However, we assume that they do not share any mutual information (e.g., shared keys). Each user is equipped with a personal device (e.g., a PDA) – referred to as a *node* – that is able to communicate via radio and to perform appropriate cryptographic operations. We also assume that each user has a unique identifier (e.g., an e-mail address) that can be equally used as a unique identifier of the user's personal device. We further assume that the nodes use moderate transmission ranges (typically less than 10 m, similar to Bluetooth). In our protocols, nodes use one-hop radio communication; that is, they do not relay packets for others when setting up shared keys. This, however, does not prevent an attacker itself from relaying packets.

We adopt the following attacker model: we assume that a third party, the attacker Mallory (M) controls the communication channel in a sense that she can eavesdrop messages and modify transmitted messages by adding her own messages to the channel. The attacker can further jam transmissions and prevent the transmission of the information contained in the message. Finally, we assume M to be computationally bounded.

We classify attackers according to their antenna gain and transmitting power. Consequently, the attacker M

[1] This is true provided that the Computational Diffie-Hellman problem [28] is intractable.

[2] We stress at this point that users could select and communicate to each other their own parameters p and g. However, this would come at the expense of the number (and size) of messages to be exchanged between the users and the complexity of the key exchange protocols.

with the highest antenna gain and transmitting power will be able to control the channel from the furthest distances. We do note however, that this space is finite given that the device's transmitting power and its receiver sensitivity are finite. Still, in our analysis, we will assume that the distance from which the attacker can control the channel is large (i.e., much larger than the devices' communication range).

3 Integrity regions

In this section, we formally introduce the concept of Integrity regions. First, we introduce the notion of the *private space*. We define it as follows:

Definition 1: A user's (device's) private space $PS(A)$ is the largest spherical space centered at user's (device's) location, within which the user can establish (e.g., visually) the presence of other wireless devices, or within which the user can assume the absence of adversaries.

In this sense, the user's (device's) private space is a space that the user (device) controls or knows. This space is typically very small (i.e., < 1 m in radius).

We now define *integrity regions* as follows:

Definition 2: An integrity region $IR(A)$ of a user A is the subspace of its private space $PS(A)$, bounded by the device's communication channel directionality.

The size of an integrity region is determined by the radius d of user's private space and depends on the directionality of device's antenna. Fig. 2 shows two examples of integrity regions, namely, devices with directional and omnidirectional antennas. Note here that the size of the integrity region is controlled by the user and is upper-bounded by the size of user's private space.

An integrity region is a user controlled space, in which the user (or a device) has sufficient confidence to establish keys.

3.1 Message Authentication & Integrity Verification

Having defined integrity regions, we now construct a *Message Transfer authenticator based on Integrity Regions* (MT-IR). With this protocol, a device can verify the integrity and the authenticity of messages received from other devices within its integrity regions.

In our description, we use the notation $\hat{x}$ to denote the message at the receiver's side when message x is sent over a public (unauthentic and insecure) channel.

Given two devices A and B, device A controls its integrity region $IR(A)$ (i.e., is aware of the presence or absence of devices). A and B choose the size of A's integrity region $IR(A)$ so that they are the only devices in this space; if they cannot ensure this at their current position, they can move to a location where this is possible. Recall that A and B do not share any secrets or hold each-others' public keys. We further assume that both devices are equipped with speakers and microphones and are able to perform ultrasonic ranging; examples of such devices are Cricket sensor motes [29].

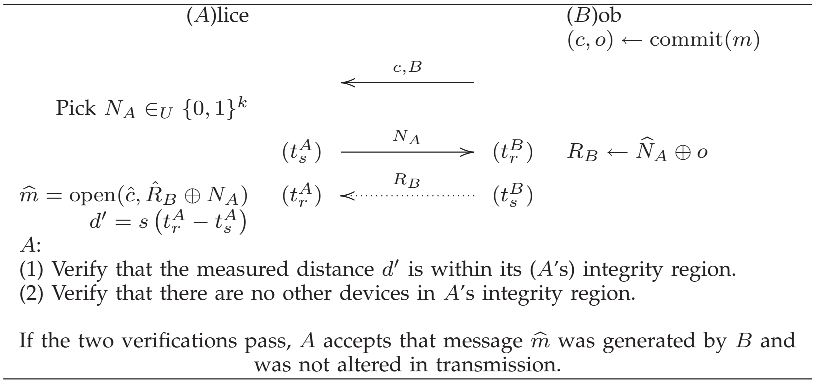
We construct our Message Transfer authenticator based on IRegions (MT-IR) as shown in Fig. 3. In this protocol, node B first commits to the message that it wants to send to A, and sends a commitment c of the message to A. As noted in Section 2.2, this commitment is both binding and hiding, and non-malleable [16]; it binds the sender to the message and hides the value of the message from the receiver. The verifier (A) then issues a fresh, randomly generated challenge N_A and measures the time until it received the response from B. This response is computed by B as a function of the challenge N_A and of the message decommitment o; it is sent over an ultrasonic communication channel and is received by A on its ultrasonic interface. After receiving this ultrasonic signal, A then calculates the distance $d' = s(t_r^A - t_s^A)$ which corresponds to the measured time of flight of the ultrasonic signal $(t_r^A - t_s^A)$; here, s is the speed of sound, and the time of flight of the radio signal is neglected. Note that N_A and R_B are exchanged bit-by-bit [30]. We do acknowledge that some time is required for processing at node B, but this time (< 100 μs) is negligible compared to the time of flight of the ultrasonic signal (approximately 2 ms for a distance of 70 cm). Node A therefore does not account for B's processing time and considers this time to be 0 s. By fixing the processing time to 0 s, we essentially turn this ranging protocol into a distance-bounding protocol on B [30], [31]. Finalizing the protocol, *(i)* A verifies that the measured distance d' is within its (A's) integrity region; this verification can be done automatically by the device if the radius d of the integrity region is known and predefined in the device (in most application scenarios, this radius can be estimated to approximately 0.3-1 m, e.g., as in implantable medical devices [4]), *(ii)* A verifies (e.g., visually) that, except for B, there are no other devices within the integrity region at any distance $d'' \leq d'$ (i.e., closer to A than B).

If both verifications pass, A accepts that the message $\hat{m}$ it received was generated by B and was not altered in transmission. In Section 3.2, we show that the vulnerability of ultrasound ranging to distance enlargement and reduction attacks does not give Mallory any advantage with respect to the MT-IR protocol. We stress at this point that A identifies B visually, and therefore associates the received message with the person (or the device) that it sees. This is illustrated in Fig. 1.

In its construction, MT-IR protocol is similar to distance bounding protocols [30], [31]. MT-IR differs from distance-bounding in that it uses range measurements within integrity regions to provide authentication and message integrity verification between devices that do not share any authentication material.

3.2 Security Analysis of the MT-IR Protocol

The MT-IR protocol achieves two goals: message authentication and integrity verification. In what follows, we analyze the security of our proposed protocol. The proofs


(A)lice
(B)ob
$(c, o) \leftarrow \mathrm{commit}(m)$
c, B
Pick $N_A \in_U \{0,1\}^k$
(t_s^A)
N_A
(t_r^B)
$R_B \leftarrow \widehat{N}_A \oplus o$
$\widehat{m} = \mathrm{open}(\hat{c}, \hat{R}_B \oplus N_A)$
(t_r^A)
R_B
(t_s^B)
$d' = s\left(t_r^A - t_s^A\right)$

A:
(1) Verify that the measured distance d' is within its (A's) integrity region.
(2) Verify that there are no other devices in A's integrity region.

If the two verifications pass, A accepts that message $\widehat{m}$ was generated by B and was not altered in transmission.

Fig. 3. Message Transfer authenticator based on Integrity Regions (MT-IR). Device A verifies if the message was received from device B and if it's integrity was preserved. (the full and the dashed arcs represent radio and an ultrasonic channel, respectively). Note that N_A and R_B are exchanged bit-by-bit.

that we provide are our adaptation of the proofs of the message transfer authenticator based on short strings comparison in [15].

Matching Conversations: We consider an active attacker Mallory in the communication model of Bellare and Rogaway [32] (for the reasoning why, refer to Section 4), meaning that Mallory can observe, modify and schedule communication between a pair of parties (A, B). In this model, a protocol $\Pi(k, I)$ is executed by a pair of parties $(A, B) \in I$, where I is a set of parties that share some common context (e.g., they all run a message authentication protocol). By $\Pi^t_{B,A}$ we mean that a party B attempts to authenticate a message from party A in a session that B believes has the session identifier $t \in \mathbb{N}$.

We let Mallory interact with $\Pi^s_{A,B}$ and $\Pi^t_{B,A}$ as oracles in a "black box" style, meaning that Mallory can query $\Pi^s_{A,B}$ by supplying A with input queries that comply to the observed authentication protocol. In the response to any query, oracle $\Pi^s_{A,B}$ outputs a message that complies to the authentication protocol. We use the following format (A, B, s, conv) to record all queries and responses that $\Pi^s_{A,B}$ sent out in the session that A marks as $s \in \mathbb{N}$; we do "the same" for $\Pi^s_{B,A}$. Here, *conv* denotes a conversation of $\Pi^s_{A,B}$, meaning a sequence of timely ordered messages that $\Pi^s_{A,B}$ has sent out and received. We say that $\Pi^s_{A,B}$ and $\Pi^t_{B,A}$ have matching conversations, if for each message m sent out by $\Pi^s_{A,B}$ in time τ_i, $\Pi^t_{B,A}$ received the same message m in τ_{i+1} and if for each message m sent out by $\Pi^t_{B,A}$ in time τ_i, $\Pi^s_{A,B}$ received the same message m in τ_{i+1} [32]. Here, $\tau_0 < \tau_1 < \tau_2 < \ldots < \tau_R$ is, for some positive integer R, a time sequence recorded by $\Pi^s_{A,B}$ and $\Pi^t_{B,A}$ when conversing.

Consider the protocol on Fig. 3; the conversations of $\Pi^s_{A,B}$ and $\Pi^t_{B,A}$ can be written as follows:

$$\begin{aligned} \mathit{conv}_B &= (\tau_0, \perp, (c, B)),\ (\tau_2, \hat{N}_A, R_B); \\ \mathit{conv}_A &= (\tau_1, (\hat{c}, B), N_A),\ (\tau_3, \hat{R}_B, \perp); \end{aligned} \tag{1}$$

where $\perp$ means that a party receives/sends no message in the corresponding time τ_i. We first observe that if the two conversations are not modified by adversary M, $\Pi^t_{A,B}$ will reach the "Accept" decision and conv_A and conv_B will be matching. This is obvious because then $\widehat{m} = \mathrm{open}(\hat{c}, \hat{R}_B \oplus N_A)$ (Fig. 3) Moreover, $\tau_0 < \tau_1 < \tau_2 < \tau_3$. This essentially means that party B will believe that the message $\hat{m}$ was sent by party A.

Definition 3: We say that $\Pi(k, (A, B))$ is a secure authentication protocol between A and B if attacker M cannot win, except with a satisfactorily small probability $O(2^{-k})$. Here, M wins if $\Pi^s_{A,B}$ and $\Pi^t_{B,A}$ reach the "Accept" decision while they do not have matching conversations.

Observe that if any of $\hat{c}$, $\hat{N}_A$, $\hat{R}_B$ are missing, $\Pi^t_{A,B}$ and $\Pi^t_{B,A}$ will simply "Abort" the protocol $\Pi(k, I)$ and adversary M will certainly fail to convince $\Pi^s_{A,B}$ to "Accept".

Security of the Authenticator: We denote the MT-IR authenticator as a protocol $\Pi(k, I)$. We observe a pair of parties $(A, B) \in I$ running $\Pi(k, I)$ and a powerful polynomially-bounded active attacker Mallory. We assume the adversary M does not belong to the set I; this is consistent with the model of Bellare and Rogaway [32] and with the fact that any two parties A and B running $\Pi(k, I)$ mutually trust each other.

In our security proof of $\Pi(k, I)$, we consider the $\mathrm{commit}(\cdot)$ function to be an ideal commitment. We further assume that each party has access to a perfect random number generator. Note that we will observe the security of $\Pi(k, I)$ in the sense of Definition 3. Let γ be the maximum number of sessions (successful or abortive) that any party can participate in. We will assume that there are at most n parties using protocol $\Pi(k, I)$. In our analysis, we will also assume that each party participates in at most one message authentication session at a time [15].

We show that for fixed A, B and t, the probability that oracle $\Pi^t_{A,B}$ outputs "Accept" without a matching conversation is satisfactorily small. Note that if $\Pi^t_{A,B}$ outputs "Accept" then there must exist some oracle $\Pi^s_{B,A}$ (with party B) that outputs "Accept" too; message i, at the end of protocol $\Pi(k, I)$, guarantees this. We first state the following intuitive result:

Lemma 1: If adversary M is to succeed against a pair of oracles $(\Pi^s_{A,B}, \Pi^t_{B,A})$, then we must have $c \neq \hat{c}$, where c is the commitment sent out by $\Pi^s_{B,A}$ and $\hat{c}$ is the commitment received by $\Pi^t_{A,B}$.

Proof: If $c = \hat{c}$ and $\Pi^s_{A,B}$ and $\Pi^t_{B,A}$ both "Accept", then $\Pi^s_{A,B}$ and $\Pi^t_{B,A}$ must have matching conversations. Indeed, M cannot break the used ideal commitment scheme, so we must have $o = \hat{o}$ and hence $m = \hat{m}$ and $N_A = \hat{N}_A$. Furthermore, since $\Pi^s_{A,B}$ "Accepts", we have $\widehat{m} = \text{open}(\hat{c}, \hat{R}_B \oplus N_A)$. Moreover, $\tau_0 < \tau_1 < \tau_2 < \tau_3$. Therefore, $\Pi^s_{A,B}$ and $\Pi^t_{B,A}$ have matching conversations. □

Consider now the interaction between a pair of oracles $(\Pi^s_{A,B}, \Pi^t_{B,A})$ and adversary M as given in (1). Assume that $(\hat{c}, \hat{o})$ is a valid commit/opening pair (i.e., M does not try to break the commitment scheme) and assume $c \neq \hat{c}$ (Lemma 1). Note that if any of the two assumptions does not hold, then M certainly fails. Then, we have the following:

Lemma 2: For any such interaction between $\Pi^s_{A,B}$ and $\Pi^t_{B,A}$ and adversary M, we have $Pr[N_A \oplus \hat{R}_B = R_B \oplus \hat{N}_A] \leq 2^{-k}$.

Proof: Observe that M has to submit $\hat{R}_B$ before actually seeing N_A. This follows from the unfolding of $\Pi(k, I)$, the hiding property of the commitment scheme and the distance measurement performed by A (as shown in Fig. 4, A verifies that B is the only device present in its integrity region hence the adversary M is located outside A's integrity region).

If M jams all traffic from B and replies on its behalf, M also has to prove that she is at distance d' where B is located. Such an attack would only be possible if the attacker would have a helper node located close to A, which would then create a radio wormhole. Since A controls its integrity region and can visually verify the absence of such attacker nodes in its proximity, this attack will fail. The only alternative for M is to therefore send a "reply" message $\hat{R}_B$ to A ahead of time, anticipating the transmission of N_A.

Thus, irrespectively of the attacking strategy taken by M, $\hat{R}_B$ is disclosed as the closing value over the unauthenticated channel. Then, we have $Pr[N_A \oplus \hat{R}_B = N_A \oplus R_B] \leq 2^{-k}$, that is, N_A and $\hat{R}_B$ are independent and uniformly distributed random variables since $\hat{R}_B$ was generated before N_A. Note that the assumption $c \neq \hat{c}$ precludes from trivial situations, where M would not modify the messages, to take place; in which case we would have $Pr[N_A \oplus \hat{R}_B = N_A \oplus R_B] = 1$. □

From Lemma 2, we conclude that the probability that there exists oracle $\Pi^t_{A,B}$ that belongs to party A and that "Accepts" without a matching conversation is at most 2^{-k} times the maximum number of interactions (successful or abortive) that party A has participated in. It is crucial that we take abortive attempts into account, too, when evaluating the probability that M is successful against a given party [15]. This is because M learns that his attempt is unsuccessful (i.e., $N_A \oplus \hat{R}_B \neq N_A \oplus R_B$) before M potentially sends out $\hat{o}$ in an attempt to disclose $\hat{m}$ to party A. If M is not successful in a given attempt, he can simply abort the protocol by simply not sending $\hat{d}$ to B.

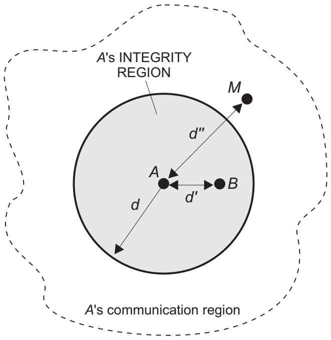


Fig. 4. Message authentication and integrity verification with MT-IR. A accepts that the message $\hat{m}$ was generated by B and was not altered in transmission only if the measured distance d' is within the integrity region ($d' \leq d$).

Since we limit each party to participate in at most γ successful or abortive runs of $\Pi(k, I)$, the probability that there exists oracle $\Pi^t_{B,A}$ that belongs to party A and that "Accepts" without a matching conversation is at most $\gamma 2^{-k}$. Note that party B "Accepts" only if the corresponding party A "Accepts". Finally, the probability that any party is broken, assuming that there are n parties that use protocol $\Pi(k, I)$, is at most $n\gamma 2^{-k}$.

3.3 Demonstration of Distance Manipulation Attacks

We implemented the MT-IR protocol using Cricket motes [29], running TinyOS. Cricket mote is a Mica-based platform [33] enhanced with an ultrasonic channel.

In our implementation, we used ON/OFF keying as the signal coding scheme for the transmission of messages over an ultrasonic channel (i.e. for the transmission of message r in the MT-IR protocol): bit "1" was encoded as a presence of the ultrasonic signal of duration 200μs within a transmission window of 60ms, and bit "0" was transmitted as the absence of signal of the same duration within the same window (60ms) (the last 200μs are considered as the start of the bit). Conforming with [30], N_A and R_B are exchanged bit-by-bit.

The commitment scheme was implemented using TinySec [34] message authentication code (MAC) implementation with Skipjack. The commit c to the message is therefore the output of the MAC function, and the decommit value o is the message itself and the secret key that was used to generate the commitment.

To better illustrate the need for a verified integrity region in our scheme, we demonstrated a range manipulation attack on the previous MT-IR implementation, which is possible if the attacker can place hidden nodes in the integrity region. By doing so, it will be able to perform a wormhole attack and pretend that it is in the

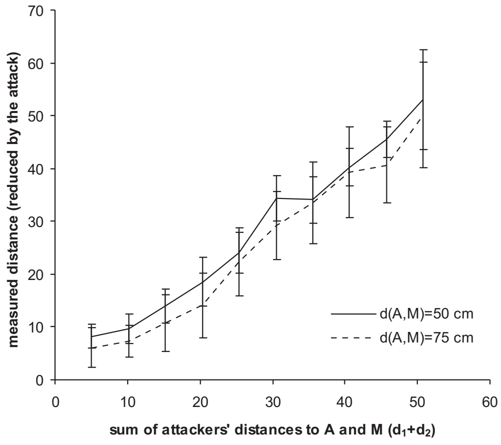


Fig. 5. Wormhole attack on ultrasonic ranging. The attacker can reduce the distance to the device A only up to the sum of the distances d_1 and d_2.

integrity region of A. Here, an attacker places node M_2 at distance $d_2 < d' < d$ from A and another node M_1 at distance d_1 from its own location.

Adversarial nodes M_1 and M_2 communicate mutually using a radio signal, through which they speed-up ultrasonic signals between M and A. Since the distance from M_1 to M can be arbitrary small (both nodes are controlled by the attacker), the only limitation that these nodes have in performing the attack is in M_2's distance to A. This can be observed in our measurement results presented in Fig. 5, which show that the attackers can successfully reduce the distance between A and M up to the value $(d_1 + d_2)$, that is, the sum of distances between M_1 and M_2 to M and A, respectively. Given appropriate control of the immediate vicinity of the devices, the attackers that are located outside the integrity region cannot violate the integrity and the authenticity of the exchanged messages.

4 Key Agreement Through Verification of Presence

In this section, we describe a key agreement protocol based on MT-IR that allows two parties that share no initial secrets (keys) or certificates to agree on a shared (secret) key.

4.1 From Secure Message Transfer Authenticator to Secure Key Agreement

In Section 3.1, we presented the message transfer authentication based on integrity regions (Fig. 3) that ensures that the message $\hat{m}$ accepted by Alice is the same message m that has been sent by Bob.

In [35], Bellare *et al.* propose a very intuitive modular approach to security analysis and construction of secure protocols. This approach assumes two adversarial models: the authenticated link model (AM) and the unauthenticated links model (UM). Roughly speaking, the AM model is an ideal-world model in which the attacker is passive (e.g., she only eavesdrops communication). On the contrary, the UM model is a real-world model, in which the attacker is active; she can replay and insert messages. The security of the protocol is first proven in the AM model, assuming (as assumed by the model itself) that all the communication between the parties is authenticated. If the protocol is proven to be secure in the AM model, then it can be shown to be secure in the UM model, provided that each message transmitted between the parties is authenticated by a MT-authenticator [35].

It is proven in [35] that the basic DH protocol is secure in the AM model, assuming that the Decisional DH problem is intractable. Now, to obtain a secure DH protocol in unauthenticated networks (UM model), we simply apply the MT-IR authenticator to each the DH public keys g^{X_A} and g^{X_B} of the basic DH protocol. Alice and Bob will run the MT-IR protocol (Fig. 3) two times; that is, *(i)* by setting $m = g^{X_B}$ and running the protocol in Fig. 3, Alice will verify the integrity and authenticity of g^{X_B}, and *(ii)* Bob will similarly verify the integrity and authenticity of g^{X_A}.

Then, based on [35], we have the following proposition:

Proposition 1: The authenticated Diffie-Hellman protocol, conditioned on the security of the MT-IR authenticator (Section 3.2), is secure (i.e., a computationally bounded adversary cannot trick the parties into accepting modified DH contributions).

However, applying the MT-IR authenticator to each DH public key results in a protocol that involves the transmission of 6 messages, two of which have the length of the shared key and need to be sent over the ultrasonic channel. Since the ultrasonic channel is much slower and less reliable than the radio channel, this might induce considerable overhead in the key agreement process.

A simple way to improve on this is to piggyback the messages of one MT-IR authenticator on the other as in the DH-IR protocol (Fig. 6): in this case, the k-bit random string N_A (sent through the commit/opening pair (c_A, o_A)) plays two roles: (1) the role of N_A and N_B. The first role is clearly fulfilled. That N_A on Fig. 6 also fulfills the second role follows from the fact that N_A on Fig. 6 remains hidden until Alice opens c_A by sending out o_A and Alice sends out o_A only after receiving $\hat{c}_B$.

4.2 The DH-IR Protocol

The optimized DH-IR protocol (described in greater detail in [5]) unfolds as shown in Fig. 6. Both Alice and Bob calculate the commitment/opening pairs ((c_A, o_A) and (c_B, o_B)) for messages $m_A \leftarrow ID_A \| g^{X_A} \| N_A$ and $m_B \leftarrow ID_B \| g^{X_B} \| N_B$, respectively. Here, N_A and N_B are k bit long random strings and "ID_A" and "ID_B" are the two public (and fixed) identifiers for devices A and B respectively, used to prevent *reflection attacks* [28] $(ID_A > ID_B)$.

In the first two messages, Alice and Bob exchange the commitments c_A and c_B. Then, in the following two

Alice			Bob
Given g^{X_A} Pick $N_A, N'_A \in_U \{0,1\}^k$ $m_A \leftarrow \mathrm{ID}_A \| g^{X_A} \| N_A$ $(c_A, o_A) \leftarrow \mathrm{commit}(m_A)$			Given g^{X_B} Pick $N_B \in_U \{0,1\}^k$ $m_B \leftarrow \mathrm{ID}_B \| g^{X_B} \| N_B$ $(c_B, o_B) \leftarrow \mathrm{commit}(m_B)$
		$\xrightarrow{c_A}$	
		$\xleftarrow{c_B}$	
		$\xrightarrow{o_A}$	$\widehat{m}_A \leftarrow \mathrm{open}(\widehat{c}_A, \widehat{o}_A)$
$\widehat{m}_B \leftarrow \mathrm{open}(\widehat{c}_B, \widehat{o}_B)$ Verify ID_B in $\widehat{m}_B$. $s_A \leftarrow N_A \oplus \widehat{N}_B$		$\xleftarrow{o_B}$	Verify ID_A in $\widehat{m}_A$. $s_B \leftarrow N_B \oplus \widehat{N}_A$
	(t_s^A)	$\xrightarrow{N'_A}$	
	(t_r^A)	$\xleftarrow{R_B}$	$R_B \leftarrow \widehat{N}'_A \oplus s_B$

$d_A = s\left(t_r^A - t_s^A\right)$, Verify $s_A \overset{?}{=} N'_A \oplus \widehat{R}_B$

Only Alice verifies her integrity region. If verification OK, Alice and Bob accept $\widehat{m}_B$ and $\widehat{m}_A$, respectively.

Fig. 6. DH-IR key Agreement Protocol.

messages they open the commitments by sending out o_A and o_B, respectively. It is important to stress that a given party opens his/her commitment only after receiving the commitment value from the other party. The first four messages are exchanged over a radio link. After receiving the commitment/opening pairs (c_A, o_A) and (c_B, o_B), Alice and Bob open the corresponding commitments and verify that "ID_B" and "ID_A" appear at the beginning of $\widehat{m}_B$ and $\widehat{m}_A$, respectively. If this verification is successful, they generate the authentication strings s_A and s_B, respectively, as shown in Fig. 6. Note that the length of each of these strings is k bits.

The main purpose of the last two messages in our protocol is to allow Alice to compare s_A against the authentication string s_B generated by Bob, in a secure way. Thus, Alice sends a k-bit long random string N'_A to Bob and measures the time until she received the response from Bob. Bob responds with $R_B \leftarrow \widehat{N}'_A \oplus s_B$, where the sign hat denotes that the N'_A as transmitted by Alice may have been altered by the adversary. Alice receives $\widehat{R}_B$; at the same time, Alice calculates the distance d_A as shown in Fig. 6 and verifies the absence of an attacker in the corresponding integrity region (see Section 3.1). If s_A equals $\widehat{R}_B \oplus N'_A$, Alice notifies Bob and they both accept the messages $\widehat{m}_A$ and $\widehat{m}_B$ (i.e., the corresponding DH public keys) as being authentic. Note that $\widehat{R}_B \oplus N'_A = s_B$ in case no attack takes place.

Security of the DH-IR Protocol: The DH-IR protocol (Fig. 6) can be divided into two stages. The first stage finishes with the calculation of the authentication strings s_A and s_B. The second stage includes the remaining two messages and the corresponding verifications.

Lemma 3: Assuming that Alice and Bob can compare s_A and s_B in a secure way, the probability that a computationally bounded adversary forges m_A and/or m_B is bounded by $O(2^{-k})$.

Proof: This proof follows from the security analysis of the MT-IR (Section 3.2). In this proof, we assume that the adversary cannot break the used commitment scheme[3] $\mathrm{commit}(\cdot)$; in practice this is possible only with a negligible probability. Let us first focus on the single run of the DH-IR protocol. Assume that the adversary tries to submit either a forged message $\widehat{m}_B \leftarrow ID_B \| g^{\widehat{X}_B} \| \widehat{N}_B$ to Alice or a forged message $\widehat{m}_A \leftarrow ID_A \| g^{\widehat{X}_A} \| \widehat{N}_A$ to Bob. Note that this is the only way for the adversary to be successful against the observed protocol run; any attacking attempt against the commitment scheme will fail (by assumption, the $\mathrm{commit}(\cdot)$ cannot be broken). We claim that in this case:

$$Pr\left[s_A = s_B\right] = Pr\left[N_A \oplus \widehat{N}_B = N_B \oplus \widehat{N}_A\right] \leq 2^{-k} \ .$$

Indeed, from the unfolding of the DH-IR protocol (Fig. 6), due to the binding, hiding and non-malleability properties of the $\mathrm{commit}(\cdot)$, the adversary has to generate and commit to strings $\widehat{N}_B$ and $\widehat{N}_A$ before actually learning messages m_A (i.e., N_A) and m_B (i.e., N_B), respectively. In other words, either N_A and N_B will be disclosed as the last value in this protocol run (i.e., the adversary will learn at least one of them after having generated and committed to $\widehat{N}_A$ and $\widehat{N}_B$). Let us assume that it is the N_A. Since N_A is a random (uniformly distributed) k-bit string, we have:

$$Pr\left[N_A \oplus \widehat{N}_B = N_B \oplus \widehat{N}_A\right] = Pr\left[N_A = \widehat{N}_B \oplus N_B \oplus \widehat{N}_A\right] \leq 2^{-k} \ .$$

Finally, for multiple runs of the protocol, the probability $Pr\left[s_A = s_B\right]$, assuming that the adversary is active, is bounded by $O(2^{-k})$.

Assuming that k is sufficiently large (i.e., 2^k is greater than the number of the protocol runs), $Pr\left[s_A = s_B\right] = 1$ will hold only when the adversary is not active (does not try to forge messages m_A and m_B). But this implies

[3] For simplicity, in this short analysis, we consider the commitment scheme to be perfect, with attacker's probability of breaking the scheme equal to $\epsilon \approx 0$. For an analysis of the impact of imperfect commitment schemes on the security of this authenticator, see [16].

that the integrity of the DH public keys g^{X_A} and g^{X_B} will be preserved. □

Similarly, the probability that a computationally bounded adversary tricks Alice into accepting $\widehat{s}_B \neq s_B$, in the second stage of the DH-IR protocol, is bounded by $O(2^{-k})$. In fact, since Alice verifies her integrity region for the presence of adversarial devices, the only option for the adversary is to try to send an appropriate value $\widehat{R}_B$ from locations outside of Alice's integrity region. However, this implies that the adversary cannot wait for the random challenge N'_A before sending back $\widehat{R}_B$. Otherwise, Alice can detect, using the measured distance d_A, that $\widehat{R}_B$ does not originate from her integrity region (see Section 3). Therefore, the only hope for the adversary is try to guess N'_A and set $\widehat{s}_B = s_A \oplus N'_{A,\text{guess}}$, where $N'_{A,\text{guess}}$ denotes the guess by the adversary.

We can conclude that by setting k to an appropriately large value, the odds of the adversary against the DH-IR protocol can be made satisfactorily small.

Optimality and Time-invariance: The DH-IR protocol involves only one k-bit long transmission over the unreliable ultrasonic communication channel. In addition, all the k bits to be communicated over the ultrasonic channel are used in the establishment of the shared secret key and therefore contribute to the uncertainty of the attacker to forge the DH public parameters. In this sense, the DH-IR protocol from Fig. 6 is optimal since *all* the transmitted bits (k-bits) contribute to the security of the shared secret key.

Concerning the time-invariance property, the number of bits k to be transmitted over the ultrasonic channel does not increase over time, even if the security parameters of the used commitment scheme $\mathrm{commit}(\cdot)$ do increase. The commitment scheme is independent of the random strings N_A, N'_A and N_B. In this sense, the DH-IR protocol from Fig. 6 is time-invariant.

5 Automatic Key Agreement using Integrity Regions

In the previous section, we showed how IRegions can be used to securely establish keys between two users. In this section, we leverage on the notion of IRegions to wireless peer-to-peer mobile networks and we propose a novel key establishment solution that is completely *automatic* and *transparent* to the user. For that purpose, we expand the integrity region's range to cover for larger distances, we drop the visual verification requirement and we leverage on users' mobility to eliminate possible attacks on the key exchange.

5.1 System Model

We assume the existence of a *profile of interest* – input by the user – that the PDA uses to filter those users with which it will set up security associations. This allows users to exclusively establish shared keys with the individuals they wish to securely communicate in the future. We say that two nodes *encounter* each other if they come within one-hop radio *communication range* of each other. We further say that such an encounter is *realized* if *(i)* the two nodes are within their respective integrity region bounds and *(ii)* the two nodes successfully run our protocols to set up shared keys. Note that two nodes can realize one encounter many times (e.g., two nodes are within communication range for a long period).

Encounters are recorded in a *table of security associations*, locally stored in each node. Keys are exchanged during the first realized encounter between two nodes. In any later realized encounter, the stored keys are reverified and used to compute a new key in such a way to preserve forward and backward security (Section 5.3). To minimize storage overhead, we only require that node i stores the *newest* established key with node j.

5.2 Main Intuition

Previous proposals in the area of key establishment (Section 2.1) are rather too "bulky" to be used for non-critical commodity applications, such as instant messaging (e.g., networks of IPhones and IPods in the city). Indeed, for such applications, if "security" impairs ease of use, then many users will simply *disregard* it [36].

Our scheme is inspired by a simple intuition that one generally establishes a shared key with someone he/she keeps on seeing and with whom he/she shares *offline* interpersonal relationship. Usually, a large number of people get together at the same place: streets, public squares, shopping malls, universities, etc.. In this way, individuals who share common interests and background "encounter" each other many times. Intuitively, as the frequency of their personal encounters increases, (i.e., their relationship ties become stronger), their level of mutual trust increases; it is exactly at this point where secure communication between these parties is required.

When two users meet for the first time, they run an adapted DH-IR key agreement protocol and verify that the exchanged messages originated from an *expected* (typically short < 10 m) distance[4] to eliminate distant MITM attacks. When both nodes *encounter* each other again, they *re-validate* and *renew* the previously exchanged key in order to rule out and *detect* possible previous attacks. Upon establishment of a shared key, the users can use it to encrypt their conversation messages. Note that a mobile attacker located in the intersection of the IRegion of both nodes (Fig. 1-a) can mount a MITM attack on the process and trick both parties into accepting a compromised key. However, we argue that our scheme limits the advantage of an adversary in mounting such attacks; indeed, during their *first* encounter, people are less likely to exchange highly sensitive information, and therefore the advantage of Mallory in acquiring this information is expected to

[4] This distance is bounded by the communication range of the devices.

be negligible. Later on, as the frequency of meetings between users increases, they tend to exchange more private information using a more robust key. Due to the absence of visual verification, Mallory could equally impersonate Bob and try to establish a shared key with Alice. However, since each user only establishes security associations with people that are listed in its profile of interest (those are the people that the user generally encounters and with whom he/she is mainly interested in communicating), this attack will be detected. We show that our scheme exhibits the *liveness property* [37] and ensures that any two *well behaved* users will *eventually* share a secure key.

5.3 Automatic Protocols for Setting up Security Associations

Our scheme for setting up a security association between two nodes (Fig. 7) consists of two sub-protocols: a *key setup* protocol and a *key verification and renewal* protocol. Both protocols are variants of the DH-IR key agreement protocol (Fig. 6) and share the same initial phase: the *advertisement phase*.

Advertisement Phase: Each node willing to set up new security associations periodically transmits *hello* messages. The purpose of the hello message is to inform any node within the sender's radio communication range of its objective to establish or verify security associations. The hello message comprises the following information:

$$\text{HELLO MESSAGE: } ID_A \| h(g^{X_A}),$$

where $\|$ denotes message concatenation, ID_A is a unique identifier of node A; h is a one-way collision resistant hash function (e.g., MD5 [38]); g^{X_A} is a freshly generated Diffie-Hellman key agreement parameter.

Note that *only* the hash of the DH key agreement parameter is sent in the advertisement phase to increase the probability of its successful reception. After receiving the hello message from node A, node B checks its table of security associations to see if it already met the initiator A. If node B finds an entry to node A in its profile of interest but did not establish any association with node A yet, it decides to run the key setup protocol with node A, otherwise it runs the key verification and renewal protocol. To prevent reflection attacks [28], we require that the initiator of the key setup (or key renewal) protocol is the node with the largest ID. That is, upon receiving an advertisement message of A, node B compares its ID with ID_A and proceeds to initiate the key exchange protocols if $ID_B > ID_A$; otherwise, node A initiates the key exchange.

Key Setup Protocol: During their first encounter, nodes A and B run a key setup protocol to establish a shared key. The key setup protocol is similar to the DH-IR protocol shown in Fig. 6. However, the two protocols differ in the fact that both the initiator A and the responder B perform range measurements to verify that the messages originated from an expected distance input by the user (typically < 10 m). This can be efficiently achieved by the use of mutual authenticated variants of the distance bounding protocol [39]. In essence, our key setup protocol relaxes the visual verification requirement of the DH-IR protocol to cover for larger IRegions radiuses. Although this would increase the advantage of nearby attackers in mounting MITM attacks, we show in Section 5.4 that our protocol ensures the authenticity and the integrity of key exchange in spite of these attackers.

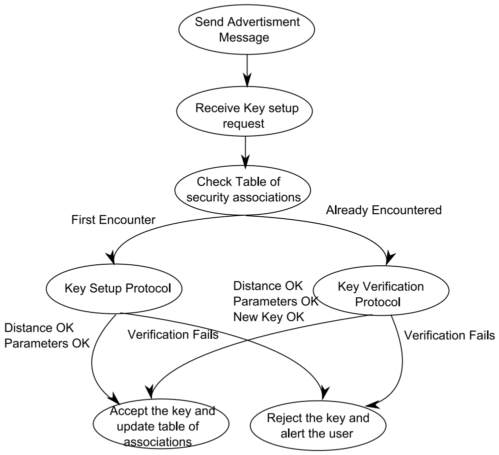

Send Advertisment Message
Receive Key setup request
Check Table of security associations
First Encounter
Already Encountered
Key Setup Protocol
Distance OK Parameters OK New Key OK
Key Verification Protocol
Distance OK Parameters OK
Verification Fails
Verification Fails
Accept the key and update table of associations
Reject the key and alert the user

Fig. 7. The Automatic Key Establishment Scheme.

If the distance verification is successful, node A generates the shared key $K_{AB} = g^{X_A X_B}$, stores it in its table of security associations and notifies B by sending the encrypted message $m'_A \leftarrow E_{K_{AB}}\{ID_A \| ID_B \| R_B\}$. Here, $E_{K_{AB}}$ denotes the encryption of the message using the shared key K_{AB}. Node B equally derives the shared key K_{AB} and decrypts m'_A to verify that ID_A, ID_B and R_B appear in the message (refer to Fig. 6 for further details). If the verification passes, it accepts key K_{AB} and stores it in its table of security associations.

Key Verification and Renewal Protocol: Our key verification protocol enables nodes to repeatedly *verify* and *renew* existing security associations to detect possible attacks, which appear as inconsistencies in shared data.

Assume that nodes A and B are about to accomplish the kth realization of their mutual encounters ($k \geq 2$). The nodes learn that this is their kth realization by comparing the nodes' advertised ID with the entries in their *local* table of security associations. From this point on, nodes A and B run the key verification protocol.

The key verification protocol is very similar to the aforementioned key setup protocol. In the first stage of the key verification protocol, nodes A and B exchange their commitments for messages $m_A \leftarrow E_{K_{AB}}\{ID_A \| g^{X_A}{}' \| N_A\}$ and $m_B \leftarrow E_{K_{AB}}\{ID_B \| g^{X_B}{}' \| N_B\}$, where $E_{K_{AB}}\{m\}$ denotes the encryption of message m using the key K_{AB}. Note that messages m_A and m_B contain new DH key agreement parameters $g^{X_A}{}'$ and $g^{X_B}{}'$ respectively. These parameters will be used to derive a new shared key.

The initiator A then compares the authentication strings (Fig. 6) and both nodes A and B verify that the messages originated from an expected distance. If these verifications pass, both nodes are *certain* that they are communicating with a node with whom they share an *authentic* secret key. Both nodes then generate a new key K_{newAB} derived from both the last established key K_{AB} and the new DH parameters contributed by each party and store it in their table of security associations. K_{newAB} is computed as follows: $K_{newAB} = F(K_{AB} \oplus g^{X_B{}' X_A{}'})$, where F is a one-way, second pre-image collision-resistant[5] pseudo-random function. In Section 5.4, we show that this scheme does not give any advantage to an attacker to disrupt key renewal provided that the last established key between A and B is indeed “authentic”. If, on the other hand, the key verification fails (i.e. there are inconsistencies in the nodes mutual histories), it is an indication that something has gone wrong (an attack has possibly taken place). The devices report this to their corresponding users, who take appropriate action.

5.4 Security Analysis

The security of our scheme relies on the assumption that any two users will *eventually* meet in the absence of an attacker in their IRegion.

Man in the Middle (MITM) Attacks: The security of our key setup and verification protocols reduces to the resilience of the DH-IR protocol (Fig. 6) against MITM attacks assuming that the attacker is not able to break the DH problem or the pseudo-random function F.

As explained in Section 4.2, the probability that a *distant* attacker (located outside the integrity region of the nodes) tricks nodes A and B into accepting an inauthentic shared key is bounded by $O(2^{-k})$, where k denotes the length of the exchanged nonces. Similarly, the probability that a computationally bounded *in-range* attacker forges the exchanged messages is equally bounded by $O(2^{-k})$.

Therefore, the only viable attack strategy for the attacker (or its compromised helper node) is to be within the IRegion of both nodes (due to the mutually authenticated distance bounding protocol and to jam all communications (including the advertisement messages) between nodes A and B. In this sense, we force Mallory to be in the IRegion of both nodes to mount these attacks on the key exchange protocols. Note, however, that Mallory cannot break the key verification protocol between nodes A and B without already having mounted a successful MITM attack on their initial key setup protocol, otherwise this attack will be detected. That is, if the attacker does not know the last key K_{AB} shared between A and B, it cannot take advantage of the key renewal process, since it cannot compute K_{newAB} (assuming that F is a perfect one-way, second preimage collision-resistant pseudo-random function).

Given this, several conditions have to be satisfied for Mallory to perform a successful MITM attack on our scheme: *(1)* Mallory has to come to the proximity of nodes A and B; *(2)* Both nodes should not have established a security association yet; *(3)* Mallory should know this; *(4)* Mallory should be able to jam all communications between both nodes; *(5)* Mallory should chase the nodes in order to prevent any possible *encounter* realization between them. However, as we show via simulations in Section 5.5, Mallory has to prevent a rather large number of encounter realizations. As a result, the cost of mounting an attack on this scheme exceeds by far the expected payoff (gain) for Mallory.

Impersonation Attacks: In this attack, Mallory takes advantage of the absence of visual verification and seeks to trick node A into believing that it is entity B.

In the case where B is present in the network, Mallory has to be sure that A and B have not realized any mutual encounter yet; otherwise, the attack will be detected. Moreover, Mallory has to prevent any possible further encounter realizations between nodes A and B. The easiest way to achieve this would be to sporadically jam any of the radio channels specific to both nodes. However, this might reveal its presence [40]. A better approach for Mallory would be to jam the messages specific to the communication between nodes A and B. In this case, Mallory should know exactly at what instant this communication happens. This means that Mallory would have to chase after nodes A or B, which makes this attack rather tedious.

On the other hand, if B happens to be absent, several conditions must be met in order to make impersonation attacks feasible: *(1)* A and B should not yet have established a security association; *(2)* Mallory should know this; *(3)* Mallory should forecast that user B will indeed be absent. Here, we assume that B is *known* to user A, otherwise A's device will not setup a security association with a user that is not listed in its profile of interest.

Assuming that all the aforementioned conditions were met, taking advantage of this attack for a considerable amount of time would be very difficult for Mallory. In fact, since B is a known user to A, we assume that A will eventually *encounter* him and this attack would be detected, unless Mallory endlessly chases both users in order to prevent any subsequent encounter realization[6].

Denial of Service (DoS) Attacks: Like most wireless communications schemes, our automatic key establishment protocol is vulnerable to signal jamming attacks by a powerful attacker. Such physical layer attacks are out of the scope of this article, strategies to detect and

[5] A function F is second pre-image collision-resistant if for a given x, it is very hard to find $x' \neq x$ such that $F(x) = F(x')$.

[6] The *sybil attack*, introduced by Douceur [41], is another way to deny service to well behaved nodes. This threat is alleviated by the use of range measurements in our key exchange protocols, which makes it be very difficult for a single attacker to perform this attack.

mitigate can be found in [7]. Another form of DoS attacks is the poisoning of the table of security associations. This attack will be detected upon the next encounter between A and B, provoking an error message.

5.5 Simulation Study

We implemented a C-based simulator to evaluate the security of our scheme. The primary goal of these simulations is to show that by exploiting users' mobility, a considerably large number of encounters (i.e., encounter realizations) per pair of users can be achieved in spite of the presence of mobile attackers.

Methodology: We performed our simulations using the restricted random waypoint model. The restricted random waypoint model is an extension of the well-known *random waypoint model* developed by Johnson *et al.* [42]. The restricted random waypoint model differs in that the choice of destination points is restricted with some probability ϕ to a fixed number of points on the specified area. This model is well suited for scenarios where people meet in public areas such as restaurants, conferences, etc.. In our simulations, we have set ϕ to be 0.6.

In our simulations, the data communication range is set to a value ranging between 2 and 10 m. Nodes can be either fixed or mobile (by default 80 % of the nodes in our setup are mobile nodes) and their movement speed can vary between 0 m/s and 2 m/s. To avoid situations in which a pair of nodes accomplishes too many realizations of a single encounter, we impose a *silence* period (of at least 30 s) between any two consecutive realizations of the same encounter by the same pair of nodes. We further assume that mobile attackers located in the IRegion radius of the communicating nodes will *always* be able to successfully perform MITM attacks on the key exchange process. We run 20 simulations of each of the aforementioned configurations[7].

Simulation Results: Our simulation results are averaged and presented with 95% confidence intervals in Fig. 8. In Fig. 8(a) and 8(b), we show the fraction of encounter realizations between all pairs of nodes with respect to the number of mobile nodes and the simulation time, respectively. For example, after 3.3 hours, more than 90% of all pairs of nodes have realized *at least one* encounter in an area of 100 m^2. This corresponds to almost 18,000 *different* encounter realizations. We further show in Fig. 8(c) the average number of encounters between all pairs of nodes; after 6 hours of simulation time, all nodes have encountered each other at least 18 times. One interesting observation here is the linear increase in the average number of encounters between all pairs with time. This is equally true as the number of mobile nodes in the system increases. This suggests that by taking advantage of users' mobility, a satisfactorily large number of encounters can be indeed achieved.

As shown in Fig. 8(e) and 8(f), almost all attacks on the key exchange process can be detected within a sufficiently small period of time (e.g., 3 hours) in spite of the presence of 25 % of malicious nodes. This further validates our analysis in Section 5.4: by exploiting node mobility, our proposed scheme can guarantee, to a large extent, that any two well behaved users will eventually share a secure key. The number of attacked encounters depends on the number of mobile nodes and the area size. As the area increases in size, it becomes more difficult for mobile attackers to predict the location of subsequent encounters between nodes (Fig. 8(d)).

We further investigated the impact of the IRegion radius on the security of our scheme. Intuitively, as the IRegion radius increases, the number of encounters between nodes increases (Fig. 8(g) and 8(h)). Therefore, although an attacker can mount MITM attacks from larger distances, this equally suggests that nodes are more likely to detect attacks on the key establishment process as the IRegion grows in size (Fig. 8(i)).

6 Discussion

In this section, we discuss how our scheme can be equally used in the exchange of asymmetric keys between users and we investigate location information leakage caused by the various advertisement messages.

6.1 Automatic Public Key Exchange

Current asymmetric cryptography based schemes (e.g., RSA [44]) are vulnerable to MITM attacks on the initial exchange of public keys between participants. Existing solutions to this problem – such as PGP [45] – require a public key infrastructure (PKI) to distribute certificates.

By relying on the notion of IRegions and the frequency of encounters between nodes, our protocols in Section 5 can be equally used to automatically authenticate and verify the exchange of public keys amongst nodes[8].

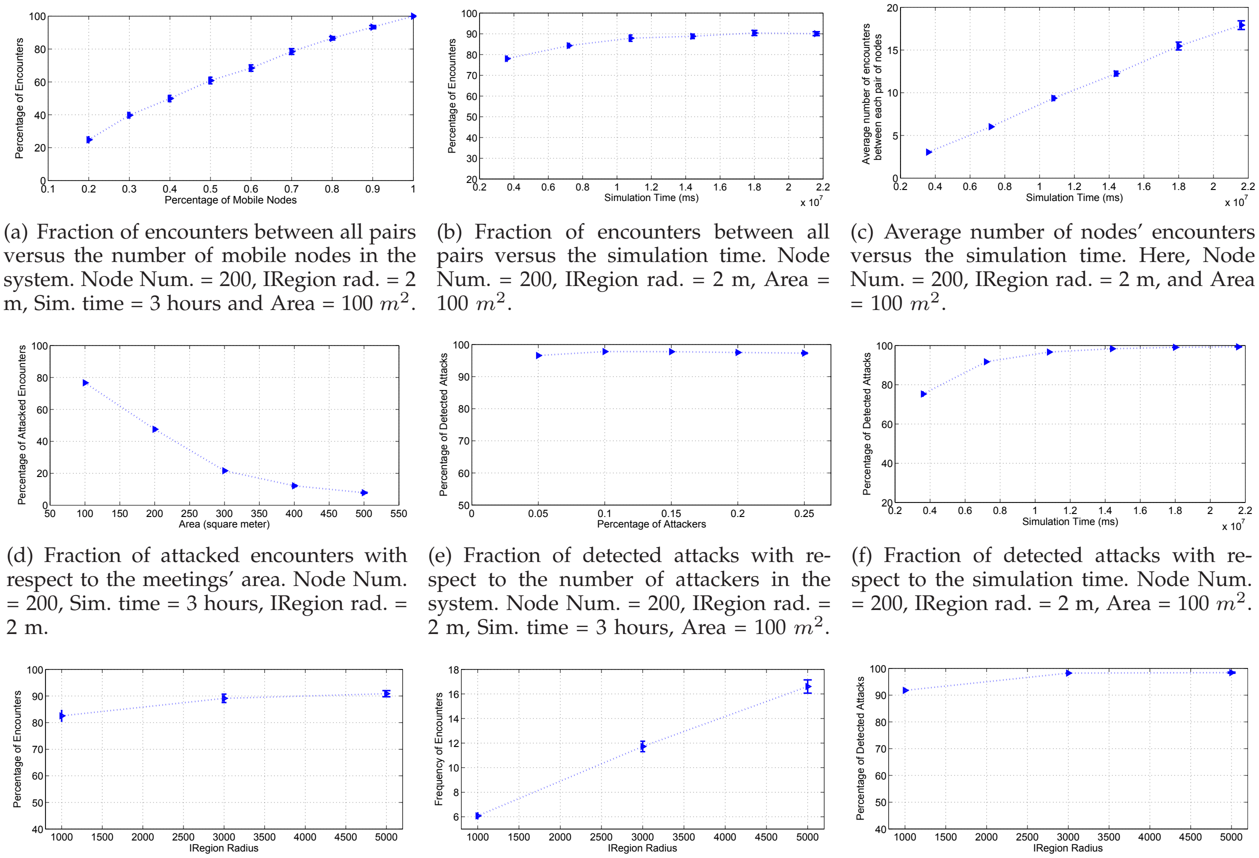
Unlike the automatic *symmetric* key protocol, by leveraging on the social network of nodes, this scheme enables additional mechanisms that can be used to prevent impersonation and MITM attacks on the public key exchange. That is, whenever a node establishes a new security association, he/she would inquire its friends about the credentials of the owner of the newly encountered node. This technique can vastly increase the number of encounters which the attacker has to manipulate.

6.2 Location information leakage

By periodically advertising HELLO messages, passive attackers can collect these announcements to gather information about the device's location. Note that this threat

[7] In addition, we conducted sample experiments using the SWIM model [43] – a novel model that proved to exhibit the same statistical properties of real mobility traces. Our results show that our scheme equally performs well given this model (we did not include the corresponding plots due to lack of space).

[8] Unlike the automatic symmetric key protocol, the basic automatic asymmetric key exchange protocol does not, however, provide forward and backward security.



(a) Fraction of encounters between all pairs versus the number of mobile nodes in the system. Node Num. = 200, IRegion rad. = 2 m, Sim. time = 3 hours and Area = 100 m^2.

(b) Fraction of encounters between all pairs versus the simulation time. Node Num. = 200, IRegion rad. = 2 m, Area = 100 m^2.

(c) Average number of nodes' encounters versus the simulation time. Here, Node Num. = 200, IRegion rad. = 2 m, and Area = 100 m^2.

(d) Fraction of attacked encounters with respect to the meetings' area. Node Num. = 200, Sim. time = 3 hours, IRegion rad. = 2 m.

(e) Fraction of detected attacks with respect to the number of attackers in the system. Node Num. = 200, IRegion rad. = 2 m, Sim. time = 3 hours, Area = 100 m^2.

(f) Fraction of detected attacks with respect to the simulation time. Node Num. = 200, IRegion rad. = 2 m, Area = 100 m^2.

(g) Fraction of encounters between pairs with respect to the IRegion Radius. Node Num. = 200, Sim. time = 3 hours, Area = 100 m^2.

(h) Average number of nodes' encounters with respect to the IRegion Radius. Node Num. = 200, Sim. Time = 3 hours, Area = 100 m^2.

(i) Fraction of detected attacks with respect to the IRegion Radius. Node Num. = 200, Sim. time = 3 hours, Area = 100 m^2.

Fig. 8. Simulation Results. Some 95% confidence intervals were omitted for better clarity.

equally applies to all wireless systems that use regular messages including an ID, e.g., WLAN, Bluetooth, etc..

In theory, the impact of ID leakage can be mitigated by making it hard to derive the name of the user from the ID and vice versa (e.g., user chosen/anonymous IDs or Zero Knowledge Protocols [46]). However, this comes at the expense of complicating the identification of nodes with which one wishes to establish keys or already has a security association. This is similar to the solution of membership protocols for *private handshakes*[47].

We propose the following solution: hello messages are modified to contain a keyed hash of the ID: $f_{h(g^{X_A})}(ID_A)\|h(g^{X_A})$, where f_x is a keyed one-way hash function with key x. We use $h(g^{X_A})$ as the key x for f_x, which could be a secure cryptographic hash function such as SHA-256. This key is then used by the recipient of the hello message to compute the keyed hashes of all IDs in the user's profile of interest. If the received ID matches any of the entries, a previous communication partner was found, and the recipient will reply with a keyed hash value of its ID. In this way, no information about the user's ID is leaked since the ID of the user cannot be derived from the keyed hash value. Note that the keyed hash value will change in every protocol instantiation. The large ID space makes exhaustive search over hashes of all keys very expensive.

7 Conclusion

Secure communication between parties remains a challenging problem owing, to a large extent, to the cumbersome key establishment between nodes. In this paper, we introduced *integrity (I) regions*, a novel security primitive that enables integrity protection of messages exchanged between entities that do not hold any mutual authentication material (i.e. public keys or shared secret keys). Integrity regions make use of lightweight ranging techniques and of visual verification within a small physical space. The proposed basic scheme effectively enables *authentication through presence* and therefore protects key establishment from man-in-the-middle attacks. We then demonstrated that integrity regions can be efficiently implemented using off-the-shelf components such as ultrasonic ranging hardware. Based on this fundamental primitive, we proposed a scheme that enables *automatic* exchange of security associations among nodes in wireless peer-to-peer networks. We further demonstrated that our proposed scheme provides a robust and *user-friendly* solution for everyday commodity applications by leveraging on physical encounters between nodes.

REFERENCES

[1] T. S. Rappaport and T. Rappaport, *Wireless Communications: Principles and Practice (2nd Edition)*. Prentice Hall PTR, December 2001.

[2] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. IT-22, no. 6, pp. 644–654, 1976.

[3] C. A. Boyd and A. Mathuria, *Protocols for Key Establishment and Authentication*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2003.

[4] K. B. Rasmussen, C. Castelluccia, T. S. Heydt-Benjamin, and S. Čapkun, "Proximity-based access control for implantable medical devices," in *CCS '09: Proceedings of the 16th ACM conference on Computer and communications security*. ACM, 2009.

[5] S. Čapkun and M. Čagalj, "Integrity Regions: Authentication Through Presence in Wireless Networks," in *Proceedings of WiSe '06*, 2006.

[6] D. Kügler, "Man in the middle attacks on bluetooth," in *Proceedings of Financial Cryptography '03*, 2003.

[7] W. Xu, W. Trappe, Y. Zhang, and T. Wood, "The feasibility of launching and detecting jamming attacks in wireless networks," in *Proceedings of MobiHoc '05*, 2005.

[8] 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.

[9] F. Stajano, *Security for Ubiquitous Computing*. John Wiley & Sons, Ltd., 2002.

[10] 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.

[11] N. Asokan and P. Ginzboorg, "Key Agreement in Ad-hoc Networks," *Computer Communications*, vol. 23, no. 17, pp. 1627–1637, November 2000.

[12] A. Perrig and D. Song, "Hash visualization: a new technique to improve real-world security," in *International Workshop on Cryptographic Techniques and E-Commerce (CrypTEC '99)*, 1999, pp. 131–138. [Online]. Available: citeseer.ist.psu.edu/perrig99hash.html

[13] D. Maher, "United states patent (no. 5,450,493): Secure communication method and apparatus," 1993.

[14] J.-O. Larsson and M. Jakobsson, "Shake. private communication with m. jakobsson."

[15] M. Čagalj, S. Čapkun, and J.-P. Hubaux, "Key agreement in peer-to-peer wireless network," in *Proceedings of the IEEE (Special Issue on Security and Cryptography)*, 2006.

[16] S. Laur, N. Asokan, and K. Nyberg, "Efficient mutual data authentication using manually authenticated strings: Preliminary version," Cryptology ePrint Archive, Report 2005/424, 2005, shorter more compact version was published at CANS 2006.

[17] C. Castelluccia and P. Mutaf, "Shake them up!: a movement-based pairing protocol for cpu-constrained devices," in *MobiSys '05: Proceedings of the 3rd international conference on Mobile systems, applications, and services*. New York, NY, USA: ACM, 2005, pp. 51–64.

[18] M. Cagalj, J.-P. Hubaux, S. Čapkun, R. Rengaswamy, I. Tsigkogiannis, and M. Srivastava, "Integrity (i) codes: Message integrity protection and authentication over insecure channels," in *SP '06: Proceedings of the 2006 IEEE Symposium on Security and Privacy*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 280–294.

[19] J. M. McCune, A. Perrig, and M. K. Reiter, "Seeing-is-believing: Using camera phones for human-verifiable authentication," in *SP '05: Proceedings of the 2005 IEEE Symposium on Security and Privacy*. Washington, DC, USA: IEEE Computer Society, 2005, pp. 110–124.

[20] M. Goodrich, M. Sirivianos, J. Solis, G. Tsudik, and E. Uzun, "Loud and clear: Human-verifiable authentication based on audio," 2005.

[21] S. Basagni, K. Herrin, D. Bruschi, and E. Rosti, "Secure pebblenet," in *Proceedings of the ACM Iternational Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc 2001*, Long Beach, CA, October 4–5 2001.

[22] R. Bobba, L. Eschenauer, V. Gligor, and W. Arbaugh, "Bootstrapping Security Associations for Routing in Mobile Ad-Hoc Networks," http://citeseer.nj.nec.com/bobba02bootstrapping.html, May 2002.

[23] J.-P. Hubaux, L. Buttyán, and S. Čapkun, "The Quest for Security in Mobile Ad Hoc Networks," in *Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC)*, Long Beach, CA, 2001.

[24] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang, "Providing Robust and Ubiquitous Security Support for Mobile Ad-Hoc Networks," in *Proceedings of International Conference on Network Protocols (ICNP)*, 2001, pp. 251–260.

[25] L. Zhou and Z. J. Haas, "Securing Ad Hoc Networks," *IEEE Network Magazine*, vol. 13, no. 6, pp. 24–30, 1999.

[26] G. Montenegro and C. Castelluccia, "Statistically Unique and Cryptographically Verifiable (SUCV) Identifiers and Addresses," in *Proceedings of the ninth Annual Symposium on Network and Distributed System Security (NDSS'02)*, San Diego, California, February 2002.

[27] G. O'Shea and M. Roe, "Child-proof Authentication forMIPv6 (CAM)," *ACM Computer Communications Review*, April 2002.

[28] W. Mao, *Modern Cryptography, Theory & Practice*. Prentice Hall, 2004.

[29] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan, "The Cricket Location-Support System," in *6th ACM MOBICOM*, August 2000.

[30] S. Brands and D. Chaum, "Distance-bounding protocols," in *EUROCRYPT '93: Workshop on the theory and application of cryptographic techniques on Advances in cryptology*. Springer, 1994, pp. 344–359.

[31] N. Sastry, U. Shankar, and D. Wagner, "Secure verification of location claims," in *WiSe '03: Proceedings of the 2nd ACM workshop on Wireless security*. ACM, 2003, pp. 1–10.

[32] M. Bellare and P. Rogaway, "Entity authentication and key distribution," in *Proceedings of the 13th Annual International Cryptology Conference*, 1993, pp. 232–249.

[33] D. Plummer, "Mica sensor platform," http://www.xbow.com.

[34] C. Karlof, N. Sastry, and D. Wagner, "Tinysec: A link layer security architecture for wireless sensor networks," in *Second ACM Conference on Embedded Networked Sensor Systems (SenSys 2004)*, November 2004, pp. 162–175.

[35] M. Bellare, R.Canetti, and H.Krawczyk, "A modular approach to the design and analysis of authentication and key exchange protocols," in *Proceedings of the 30th Annual Symposium on the Theory of Computing*, 1998.

[36] R. Anderson, H. Chan, and A. Perrig, "Key infection: Smart trust for smart dust," in *12th IEEE International Conference on Network Protocols (ICNP 2004)*, 2004, pp. 206–215.

[37] N. A. Lynch, *Distributed Algorithms*. Morgan Kaufmann Publishers, Inc, 1996.

[38] R. L. Rivest, "The MD5 message digest algorithm," RFC 1321, Internet Activities Board, Internet Privacy Task Force, April 1992.

[39] S. Capkun, L. Buttyn, and J. P. Hubaux, "SECTOR: Secure Tracking of Node Encounters in Multi-hop Wireless Networks," in *Proceedings of the 1st ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN)*, October 2003.

[40] A. Wood and J. Stankovic, "Denial of service in sensor networks," *Computer*, vol. 35, no. 10, pp. 54–62, Oct 2002.

[41] J. Douceur, "The Sybil Attack," in *Proceedings of the IPTPS02 Workshop*, Cambridge, MA (USA), 2002. [Online]. Available: citeseer.nj.nec.com/douceur02sybil.html

[42] D. Johnson and D. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," in *Mobile Computing*, edited by T. Imielinski and H. Korth, chapter 5, pages 153-181. Kluwer Academic Publishers, 1996.

[43] A. Mei and J. Stefa, "SWIM: A Simple Model to Generate Small Mobile Worlds," in *Proceedings of IEEE INFOCOM '09*, Rio de Janeiro, Brazil, April 2009.

[44] R. L. Rivest, A. Shamir, and L. M. Adelman, "A method for obtaining digital signatures and public-key cryptosystems," Tech. Rep. MIT/LCS/TM-82, 1977.

[45] P. R. Zimmermann, *The Official PGP User's Guide*. Cambridge, MA, USA: MIT Press, 1995, iSBN 0-262-74017-6.

[46] S. Goldwasser, S. Micali, and C. Rackoff, "Knowledge Complexity of Interactive Proof Systems," in *Proceedings of STOC '85*, 1985, pp. 291–304.

[47] J.-H. Hoepman, "Private handshakes," in *Proceedings of ESAS*, 2007, pp. 31–42.

Srdjan Čapkun is an Assistant Professor in the Department of Computer Science, ETH Zurich. He received the Dipl. Ing. degree in Electrical Engineering / Computer Science from University of Split, Croatia in 1998, and the Ph.D. degree in Communication Systems from EPFL (Swiss Federal Institute of Technology - Lausanne) in 2004. Prior to joining ETH he was a postdoctoral researcher in the Networked & Embedded Systems Laboratory (NESL), University of California Los Angeles and an Assistant Professor in the Informatics and Mathematical Modeling Department (IMM), Technical University of Denmark (DTU). His research interests include the design and the analysis of security protocols for wireless and wireline networks. He is an associate editor of IEEE Transactions on Mobile Computing and an area editor of ACM MC2R. He was a program chair of the ACM Conference on Wireless Network Security (WiSec), 2008. He is a member of the Zurich Information Security Center (ZISC) and of the RFID Consortium for Security and Privacy (RFID-CUSP). He is a member of the ACM and IEEE.

Mario Čagalj received the Dipl.Ing degree in computer science and electrical engineering from the University of Split, Croatia, in 1998, and the PhD degree in communication systems from the Ecole Polytechnique Federale de Lausanne (EPFL) in February 2006. In 2000 and 2001, he completed the Predoctoral 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. Since September 2006, Mario Cagalj is an Assistant Professor at Faculty of Electrical Engineering, Mechanical Engineering, and Naval Architecture (FESB), University of Split, Croatia. His research interests include the design and analysis of security protocols for wireless networks, applied cryptography, applications of game theory to wireless (and wired) networks, and the design of energy-efficient communication protocols for wireless networks. More details can be found at http://www.fesb.hr/~mcagalj. He is a member of the IEEE.

Ghassan Karame holds a Bachelor in Computer and Communications Engineering from the American University of Beirut (AUB). He completed his Masters of Science in Information Networking from Carnegie Mellon University (CMU) in December 2006. Since December 2006, he worked as a graduate researcher in the Autonomic and Grid Computing Group of Athens Information Technology, where he was involved in the design and implementation of a 2D Face Tracking system in the context of EU project CHIL. In April 2007, he joined the System Security Group of ETH Zurich as a PhD student. His research interests include security in distributed systems, secure service verification protocols, cryptographic puzzles, security in reputation-base systems, among many others. He is a student member of the IEEE.

Nils Ole Tippenhauer received the Dipl. Ing. degree in Computer Engineering from the TUHH in Hamburg, Germany, in 2007. After graduation, he joined the System Security group at ETH Zurich where he is currently working towards a PhD degree. His current research is directed towards the implementation of secure ranging (distance bounding and authenticated ranging). Other interests include security analysis of existing hard- and software in mobile and embedded devices. He is a student member of the IEEE.