

Random Key Assignment for Secure Wireless Sensor Networks*

Roberto Di Pietro, Luigi V. Mancini, and Alessandro Mei
Dipartimento di Informatica
Università di Roma "La Sapienza"
Via Salaria 113, 00198-Roma, Italy.
{dipietro,mancini,mei}@dsi.uniroma1.it

ABSTRACT

A distributed Wireless Sensor Network (WSN) is a collection of n sensors with limited hardware resources. Sensors can exchange messages via Radio Frequency (RF), whose range usually covers only a limited number of other sensors. An interesting problem is how to implement secure pair-wise communications among any pair of sensors in a WSN. A WSN requires completely distributed solutions which are particularly challenging due to the limited resources and the size of the network. Moreover, WSNs can be subject to several security threats, including the physical compromising of a sensor. Hence, any solution for secure pairwise communications should tolerate the collusion of a set of corrupted sensors. This paper describes a probabilistic model and two protocols to establish a secure pair-wise communication channel between any pair of sensors in the WSN, by assigning a small set of random keys to each sensor. We build, based on the first Direct Protocol, a second Co-operative Protocol where the assured level of security can dynamically change during the life-time of the WSN. Both protocols also guarantee implicit and probabilistic mutual authentication without any additional overhead and without the presence of a base station. The performance of the Direct Protocol is analytically characterized while, for the Co-operative Protocol, we provide both analytical evaluations and extensive simulations. For example, the results show that, assuming each sensor stores 120 keys, in a WSN composed of 1024 sensors with 32 corrupted sensors the probability of a channel corruption is negligible in the case of the Co-operative Protocol.

Keywords

Distributed wireless sensors networks, key management protocols, secure pair-wise communications, sensor-to-sensor authentication.

1. INTRODUCTION

*This work was partially supported by the EU under the IST-2001-34734 EYES project, by the WEB-MINDS project supported by the Italian MIUR under the FIRB program, and by the "Progetto Giovani Ricercatori" program from the Italian MIUR.

A Wireless Sensors Network (WSN) is a collection of sensors whose size can range from a few hundred to a few hundred thousands and possibly more sensors. The sensors do not rely on any pre-deployed network architecture, they thus communicate via an ad-hoc wireless network. Distributed in irregular patterns across remote and often hostile environments, sensors should autonomously aggregate into collaborative, peer-to-peer networks. Sensor networks must be robust and survivable despite individual sensor failures and intermittent connectivity (due, for instance, to a noisy channel or a shadow zone). WSNs are often infrastructure-less, and the power supply of each individual sensor is provided by a battery, whose consumption for both communication and computation activities must be optimized.

A WSN can be deployed in both military and civil scenarios [15, 18]. For instance, it could be used to (1) provide a relay network for tactical communication in a battlefield; (2) collect data from a field in order to reveal the presence of a toxic gas; (3) facilitate rescue operations in wide open hostile areas; (4) fulfill perimeter surveillance duties; (5) operate for commercial purpose in severe environmental constrained scenarios (for instance, to measure the concentration of metals such as nodules of manganese on the ocean bed). Moreover, a WSN can be used to enforce physical access control by checking secure access to a building. Each person should carry a sensor which contains some sort of encrypted personal information. This information is exchanged with other sensors distributed across the building, and can be used to authenticate, to assign the appropriate clearance to the users, and to trace their movements in the building.

Establishing secure pair-wise communication can be useful for many application. In particular, [14] shows this is a pre-requisite for the implementation of secure routing. Also, it can be useful for the establishment of secure group communications as well. The naïve technique for implementing secure pair-wise communication is to assign a set of secret keys to each component of the group, each key of the set being shared with only one other member of the group. This solution requires each member to store $n - 1$ keys, where n is the size of the group, with $\frac{n(n-1)}{2}$ different keys in the group. In this paper, we focus on confidentiality, and provide models and protocols to allow any pair of sensors of the WSN to establish a confidentially secure communication channel (*secure channel* in the following) while loading each sensor with a small, constant size set of keys. We devise a first protocol, the Direct Protocol, which establishes a pair-wise communication channel that is secure with a fixed probability. We then build on this protocol a second one that allows to trade off the desired level of security with the protocol

overhead. Note that the overhead for establishing the communication channel is paid only once for any channel set up. The security of both approaches is analytically described, and these results have been validated with extensive simulations. We also show that both protocols guarantees implicit and probabilistic mutual sensor to sensor authentication, with excellent scaling properties.

In Section 2, we report on the related work in the literature. In Section 3, we define our system assumptions. In Section 4, we introduce the Direct Protocol for the establishment of a secure pairwise communication channel between any two sensors. In Section 5, we describe the Co-operative Protocol, which is adaptive to the threat, i.e. the number of corrupted sensors in the WSN. We achieve this goal with an increase in the communication overhead. Note that this is incurred only once for every channel establishment. In Section 6, we show that both protocols provide probabilistic sensor to sensor authentication. In Section 7, we discuss our analytical and experimental results, while Section 8 presents some concluding remarks.

2. RELATED WORK

Several recent research works focus on key establishment protocols in dynamic peer groups. In particular, [23] deals with the problem of key agreement in dynamic peer groups and recognizes that key agreement, especially in a group setting, is the stepping stone for all the other security services. The paper also presents a concrete protocol suite, CLIQUES, which offers complete key agreement services. CLIQUES is based on multi-party extensions of the well-known Diffie-Hellman key exchange method [9]. The protocols are provably secure against passive adversaries. In [1], the above protocols are enhanced to provide services like authenticated key agreement in dynamic peer groups with the emphasis on efficient and provably secure key authentication, key confirmation and integrity. However, the protocols in [1] and [23] use public key cryptography. As it was pointed out in [19], public key cryptography is not well-suited for securing WSNs. Indeed, the memory of a sensor is typically insufficient to hold the long keys necessary to guarantee secure asymmetric cryptographic. Moreover, sensors are usually equipped with a low power processor which requires too long and too much energy to compute the modular exponentiations involved in the implementation of public key cryptography.

Among research specifically focused on WSNs, [19] shows how to implement a security subsystem for a limited wireless sensor network platform. The system assumes the presence of base stations acting as gateways for inter-sensor communications. Being a base station more powerful (we can think of it as a workstation), it is reasonable to assume it can easily implement public key cryptography. By assuming the base stations trusted computing bases, [19] can relax the assumption of tamper-resistance of sensors. The security subsystem is shown to support a few security functions, such as authenticated and confidentiality communications as well as authenticated broadcast. In [3], a key management scheme is proposed which periodically updates the symmetric keys employed by the sensors. However, neither forward nor backward secrecy is guaranteed. In [5] the focus is the assessment of the overhead introduced by key management on power consumption. Different schemes for key establishment in WSN are examined and a couple of new schemes are proposed.

In [10], the idea of probabilistic key sharing for WSN is firstly introduced and the authors provide a centralized algorithm for re-keying in a distributed WSN. Recently and independently from this

work, both [6] and [25] present protocols based on random assignment of keys in a WSN. In [25] a distributed group key management protocol for mobile ad hoc networks is proposed. The protocol supports two operations: group re-keying and threshold based sensor revocation. The group re-keying scheme, as well as the threshold based sensor revocation, are based on probabilistic key sharing. In [6], three new mechanisms in the framework of random key predistribution to address the bootstrapping problem are proposed. First, the *q-composite* random key predistribution scheme achieves probabilistic security under small scale attack while trading off increased vulnerability in the face of a large scale physical attack on network sensors. Second, the multi-path key reinforcement scheme substantially increases the security of key setup such that an attacker has to compromise many more sensors, with respect to the *q-composite* scheme, to achieve a high probability of compromising any given communication. Finally, the random-pairwise keys scheme assures that, even when part of the sensors have been compromised, the rest of the network remains fully secure. However, there are many differences between the contributions in [6] and in this work, namely: (1) the *q-composite* scheme is less energy efficient than the scheme detailed in Section 4, in particular because of the key discovery phase which requires a number of messages proportional to the number of keys assigned to each sensor; (2) the multi-path reinforcement scheme requires the availability of disjoint paths between the sender and the receiver. Disjoint multi-path finding is an NP-hard problem, and moreover requires knowledge of the topology. We devise a completely different solution in Section 5; and (3) the sensor-to-sensor mutual authentication scheme of [6] does not scale. Our probabilistic authentication mechanism has excellent scaling properties and is adaptive to the level of confidence required by the authentication process.

3. SYSTEM ASSUMPTIONS

Since sensors communicate using radio frequencies (RF), broadcasting is the fundamental communication primitive. Each sensor can directly communicate only to a limited number of other sensors, those within its communication range, and packets are delivered to destination via multi-hop. Moreover, as it has been shown in [13], communication is the main power-demanding operation. Therefore, communications should be maintained to the strict necessary. Note that sending a message consumes more energy than receiving a message [22], and this is a parameter of interest in designing a key establishment protocol.

3.1 Sensor Components

This section describes the components of each sensor required to implement our protocols. Note that these components can be independent modules to be added to the basic standard components. Each sensor can execute a one-way hash function, \mathcal{H} in the following, a symmetric encryption algorithm E , a keyed hash $HMAC(key, value)$, and a pseudo-random number generator. None of these modules need to be kept secret. Moreover, we assume that each sensor has enough memory to store a limited number of keys. Finally, note that we do not need to assume that sensors in the WSN are tamper-resistant, thus relaxing the assumptions in [3, 17, 8].

In the following, $\mu' = E_{\kappa}(\mu)$ is an encrypted message where μ is the plain-text, κ is the secret key, and E is the encryption algorithm. Accordingly, $\mu = E_{\kappa}^{-1}(\mu')$ is the decryption of the same message. A summary of the notation employed throughout this paper is reported in Table 1.

3.2 The Threat Model

Table 1: Summary of the notation.

a, b	two generic sensors in the WSN;
\mathcal{N}	set of sensors in the WSN;
n	number of sensors in the WSN; $n = \mathcal{N} $;
P	size of the pool from which the keys are drawn;
k	the number of keys assigned to each sensor;
v_a^i	i^{th} key assigned to sensor a
V_a	set of keys assigned to sensor a ; $V_a = \cup_{i=1}^k \{v_a^i\}$
C	the set of co-operating sensors;
m	number of co-operating sensors; $m = C $;
\mathcal{W}	set of corrupted sensors in the WSN;
ω	number of corrupted sensors in the WSN; $\omega = \mathcal{W} $;
ω_i	i^{th} corrupted sensor in \mathcal{W} ;
W	set of all corrupted keys; $W = \cup_{i=1}^{\omega} \{V_{\omega_i}\}$;
E	encryption function;
\mathcal{H}	hash function;
\oplus	xor operation.

The attacks considered in this paper are: (1) *passive attacks*; (2) *active attacks*. In passive attacks, we assume that an eavesdropper can constantly monitor the whole WSN. We consider two types of passive attacks that an adversary can perform: (1) cipher text attack, that is, given the cipher text, the adversary tries to recover the encryption key; and (2) chosen plain text attack, that is, the adversary can feed the sensor with known data and then observe the encrypted message sent by the sensor. Therefore, we consider confidentiality and authenticity of data of paramount importance. In *active attacks*, we assume the attacker can *capture* a sensor, collecting all the information and the keys the sensor is loaded with. Moreover, we consider a worst case scenario, that is we assume that all the compromised sensors in the WSN are compromised by the same attacker and thus collude to compromise the network. Finally, we assume that the environment in which the sensors operate is untrusted. Each sensor trusts itself, while sensors do not trust each other.

4. THE DIRECT PROTOCOL

The constraints that make the task of securing a channel between any pair of sensors not trivial include: a limited amount of memory and a limited battery power available to each sensor. Henceforth, we want to design a protocol that is: (1) memory efficient: only a limited amount of memory is required to store crypto keys; (2) energy efficient: low computation and communication overhead; (3) resilient to the coalition of ω *corrupted sensors*. To match these requirements, we will provide:

- a *key deployment scheme* describing how the sensors are loaded with the keys;
- a *key discovery procedure* which enables an arbitrary pair of sensors to compute the set of keys they share;
- a *security adaptive channel establishment procedure*, a mechanism to enable an arbitrary pair of sensors to agree on a common key to be used to secure the channel.

4.1 The Key Deployment Scheme

Assume an n sensor Wireless Sensor Network. The random key pre-deployment strategy proposed in [10] is composed of the following steps:

1. a *pool* of P random keys $\{v_p^1, \dots, v_p^P\}$ is generated;

2. for each sensor a in the WSN, a set $V_a = \{v_a^1, \dots, v_a^k\}$ of k distinct keys is randomly drawn from the pool and assigned to a .

If the above key deployment scheme is used, the corresponding channel establishment procedure could be quite time and energy consuming. Even if two sensors a and b are in their communication range and share some keys, to discover which keys they actually share is not efficient. Indeed, sensor a is supposed to broadcast messages $\alpha, E_{v_a^i}(\alpha), i = 1, \dots, k$ where α is a challenge. The decryption of $E_{v_a^i}(\alpha)$ with the proper key by sensor b would reveal the challenge α and the information that b shares that key with a . This *key discovery* procedure requires k^2 decryptions on the receiver side and k encryptions on the sender side. Moreover, at least k messages have to be sent and received.

As observed in [25], a pseudo-random key deployment scheme allows a more efficient key discovery procedure than a random scheme. Given a sensor a , the idea is to generate the indexes of the keys that will be assigned to a pseudo-randomly. The generator is initialized with a publicly known seed dependent on a . Once the seed is known, the k indexes of the keys assigned to a can be computed by anyone. Note that this key discovery procedure reveals only the indexes of the keys given to sensor a , and does not leak any information on the keys themselves. The above pseudo-random method requires a limited amount of additional storage per sensor in comparison with the key assignment in [10]. Indeed, each sensor has to store, along with the keys, also the index of each key. However, the new key discovery procedure now requires no message exchange, at most k applications of the pseudo-random generator, and k look-ups in the local memory.

In the following we adopt this pseudo-random, *seed-based* key deployment strategy, and for this strategy we are interested in evaluating its main properties: (1) the channel establishment effectiveness between any two sensors; (2) the efficiency of the channel establishment procedure; (3) the resilience of the channel against node capture.

4.1.1 Channel Existence

Given two sensors a and b loaded with sets of keys V_a and V_b according to the seed-based strategy, it is important to assess the probability that a channel exists between two sensors in the WSN.

DEFINITION 1. A communication channel (channel in the following) exists between sensors a and b if and only if $V_a \cap V_b \neq \emptyset$, that is a and b share at least one key.

From Definition 1, the probability that a channel exists between sensors a and b is equal to the probability that a and b share at least one key of the pool. Let E be the event: "There is at least one key shared by a and b ". $\Pr[E]$ can be directly computed as:

$$\Pr[E] = 1 - \Pr[\bar{E}] = 1 - \frac{\binom{P-k}{k}}{\binom{P}{k}}. \quad (1)$$

Note that channel existence scales with the WSN size. Indeed, the probability of channel existence is independent from n , being dependent only on k and P . In Figure 1 we plot the probability of channel existence between sensors a and b as a function of the per

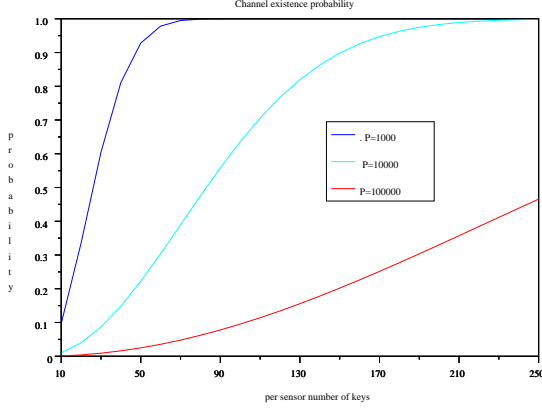


Figure 1: Channel existence probability for different values of P and k .

sensor number of keys k and for three values of the pool size P . It is remarkable that, even with a small number of keys loaded on the sensors, the channel has high probability of existence.

4.2 The Protocol

Assume a channel exists between sensors a and b and that a wants to securely communicate with b . The protocol we propose in Table 2 will provide both sensors with a session key $k_{a,b}$ that will be used to secure the channel. The proposed protocol is as follows: first, a computes all the keys it shares with b (this is always possible since the indexes of the keys of b can be pseudo-randomly computed by anyone from the publicly known seed). Then, the keys are XOR-ed together. The resulting value

$$k_{a,b} = \bigoplus_{v_a^i \in V_b} v_a^i \quad (2)$$

will be used to secure all messages between a and b . Sensor b , once it receives the first message from a , builds $k_{a,b}$ in the same manner and it is thus able to decrypt the message. Note that $k_{a,b} = k_{b,a}$, hence, the session key does not depend on which of the two sensors initiates the communication. In the first message only, sensor a will place a known value (*key_est* in Table 2) to allow the receiver to check that session key $k_{a,b}$ has been correctly computed. In case sensor b agrees on $k_{a,b}$, sensor b replies by sending to a , encrypted with key $k_{a,b}$, value *key_est_confirm* to confirm the establishment of the channel.

In the construction of key $k_{a,b}$, our protocol employs all the shared keys between sensors a and b . Intuitively, $k_{a,b}$ is as strong as the strongest key sensors a and b can agree on. The attacker has to know all of the keys shared between the two sensors to corrupt the channel. In the following we will often refer to the shared key $k_{a,b}$ as the channel (indeed $k_{a,b}$ provides the effective implementation of the channel). Expressions like "the channel has been corrupted" means that the coalition of the corrupted sensors has been able to recover all the shares of key $k_{a,b}$.

The protocol is described in detail in Table 2. Note that the resulting value of variable *found* is used by sensor a to check whether the channel with b exists or not. In the case there is not a channel between a and b , $k_{a,b}$ is equal to 0 and the Direct Protocol fails.

Table 2: Pair-wise probabilistic secure communication channel establishment

```

Direct_Protocol(b: sensor.ID)
Input: the ID of the receiver sensor;
Output:  $k_{a,b} = \bigoplus_{v_a^i \in V_b} v_a^i$ 

1.  $k_{a,b} = 0$ ;
2. dest_indexes = key_discovery( $b$ );
3. found = false;
4. for all dest_ix  $\in$  dest_indexes
5.   if dest_ix  $\in$  my_key_indexes then begin
6.      $k_{a,b} = k_{a,b} \oplus \text{my\_keys}[\text{dest\_ix}]$ ;
7.     found = true;
8.   end
9. if NOT(found) then error
10. else  $a \rightarrow b : < a, b, E_{k_{a,b}}(\text{key\_est}) >$ ;

```

Assume that the channel between sensors a and b is under the attack from a coalition $\mathcal{W} = \{\omega_1, \dots, \omega_\omega\}$ of $\omega > 0$ corrupted sensors. The corrupted sensors know W , the set of all the keys they have been loaded with:

$$W = \bigcup_{i=1}^{\omega} V_{\omega_i}$$

We focus on event $C_{a,b} \equiv$ "all the keys shared by a and b are known by the set \mathcal{W} of the corrupted sensors", that is $W \supseteq V_a \cap V_b$. Event $C_{a,b}$ can be written as a combination of events $E_i \equiv$ "key v_a^i either is in W , or is not in V_b ":

$$C_{a,b} \equiv \bigwedge_{i=1}^k E_i \equiv \bigwedge_{i=1}^k (v_a^i \in W \vee v_a^i \notin V_b)$$

Unfortunately, while $\Pr[E_i]$ is easy to compute, $\Pr[C_{a,b}]$ is still not easy since events E_i are *not* independent. We go around this problem by focusing on $\bar{C}_{a,b}$, which occurs if at least one of the keys shared by a and b is not present in set W . Hence:

$$\Pr[\bar{C}_{a,b}] = \Pr\left[(v_a^1 \in V_b \wedge v_a^1 \notin W) \vee \dots \vee (v_a^k \in V_b \wedge v_a^k \notin W) \right]$$

Let $A_j = (v_a^j \in V_b \wedge v_a^j \notin W)$. Applying the formula to compute the probability of the union of k events, we get:

$$\Pr[\bar{C}_{a,b}] = \sum_{s=1}^k (-1)^{s+1} S_s$$

where $S_1 = \sum_{s=1}^k \Pr[A_s]$; $S_2 = \sum_{s < t} \Pr[A_s \wedge A_t]$; \dots ; $S_k = \Pr[\bigwedge_{j=1}^k A_j]$; that is S_h is the sum, taken over all collections of precisely h events, of the probabilities that all these h events occur [12]. The generic

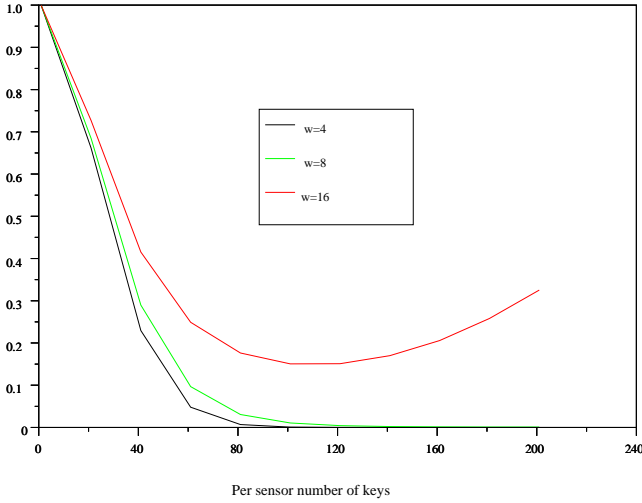


Figure 2: Corruption probability of the channel between a and b for $P = 1000$ and $\omega = 4, 8, 16$, as k increases.

term S_i can be written as:

$$S_i = \binom{k}{i} \Pr \left[\bigwedge_{j=1}^i A_j \right] = \binom{k}{i} \Pr \left[\left(\bigwedge_{j=1}^i (v_a^j \in V_b) \right) \wedge \left(\bigwedge_{j=1}^i (v_a^j \notin W) \right) \right] = \binom{k}{i} \frac{\binom{P-i}{k-i}}{\binom{P}{k}} \left(\frac{\binom{P-i}{k}}{\binom{P}{k}} \right)^\omega.$$

According to the above equations:

$$\Pr [C_{a,b}] = 1 - \Pr [\bar{C}_{a,b}] = 1 - \sum_{i=1}^k (-1)^{i+1} \binom{k}{i} \frac{\binom{P-i}{k-i}}{\binom{P}{k}} \left(\frac{\binom{P-i}{k}}{\binom{P}{k}} \right)^\omega. \quad (3)$$

Figure 2 shows the probability that a channel between two sensors is corrupted by \mathcal{W} , for different values of parameters k , P and ω . For small values of k , the qualitative behavior of the curves confirms the initial intuition that increasing the number of keys loaded on each sensor enhances the security of the channel. However, this intuition is not completely correct. Indeed, the bigger is the set of keys assigned to each sensor, the bigger the set of different keys the corrupted sensors can collect. This latter effect is overwhelming when k gets larger. This behavior is evident in Figure 2 from the curve of parameter $\omega = 16$, but it is present for any value of ω . Observe that, if $k = P$, then the probability of channel corruption is 1.

5. THE CO-OPERATIVE PROTOCOL

5.1 The Protocol

As it has been shown in Section 4, the Direct Protocol allows two sensors to communicate with a certain degree of security. The degree of security achieved depends on parameters k , P and ω , according to Equation 3. The main drawbacks of the Direct Protocol are the following:

- the Direct Protocol is not adaptive to changes in the threat (for instance, the number of corrupted sensors overcomes the value ω assumed as an upper bound in the design project of

the WSN), or in the security requirements (for instance, a higher security level is desired due to the temporary management of more sensitive information).

- for a fixed sensor key ring size k , the probability of channel corruption can be not satisfactory even for small values of the number of corrupted sensors (for instance, see Figure 2 for 16 corrupted sensor).

To overcome the above drawbacks, we propose a scheme where, with an increase in the communication overhead, the key establishment phase is made co-operative. If sensor a wants to establish a secure channel with sensor b , sensor a chooses a set $C = \{c_1, \dots, c_m\}$ of co-operating sensors such that $a, b \notin C$ and $m \geq 0$. Then, a sends a request of co-operation to each of the sensors in C . The request carries the ID of b . Each sensor c in C transforms its original channel key with b , $k_{c,b}$ as follows: first, $k_{c,b}$ is built according to the Direct Protocol; then, a share is created by hashing $k_{c,b}$ with the id of a

$$HMAC(ID_a, k_{c,b}); \quad (4)$$

finally, the share is sent to a . When all the shares are received, sensor a computes $k_{a,b}$ and combines it with all the shares to obtain a cooperative channel key $k_{a,b}^C$

$$k_{a,b}^C = k_{a,b} \oplus \left(\bigoplus_{c \in C} HMAC(ID_a, k_{c,b}) \right). \quad (5)$$

Sensor a sends $\mathcal{H}(k_{a,b}^C)$ to b along with the list of sensors in C . Sensor b , once it has received this message, has all the information required to locally compute $k_{a,b}^C$, without sending or receiving any other message, and to double-check the resulting key with $\mathcal{H}(k_{a,b}^C)$. Note that, when $m = 0$, the Direct and Co-operative Protocol are equivalent.

Table 3 shows in detail the pseudo code of the Co-operative Protocol. In Figure 3, we show a simplified instance of the messages exchanged by the Co-operative Protocol in the case there is only one co-operating sensor (c_1). In step (1), sensor a sends the request of co-operation to sensor c_1 ; once c_1 has received such a request, it computes $k_{c_1,b}$ according to the Direct Protocol and sends a transformed and not-invertible image of such a value to a , as shown in step (2). Finally, sensor a computes $k_{a,b}^C$ according to Equation 5, and sends this value to b , together with the set of sensors (c_1) that co-operated in building the channel (step (3)). Note that it is mandatory for sensor a to send, in step (3), the list of sensors in C , otherwise b would not be able to compute $k_{a,b}^C$. Such a list is at most $m \log n$ bits long.

Set C can be chosen according to several policies. A first energy preserving option is to include only relatively nearby sensors in C . For example, only sensors within one or two hops. This choice yields a more efficient key setup phase, though the protocol can be weaker against geographically localized attacks. A second option is to choose C randomly. At the price of a larger communication overhead especially in large networks, the protocol would support both random and geographically localized attacks. Third, sensors can be chosen according to individual properties, like tamper-resistance, giving a potentially more secure channel. Of course, mixtures of the above policies are also possible.

The Co-operative Protocol shows the following features:

Table 3: Pseudo code of the Co-operative Protocol for the pairwise key establishment

<p>Co-operative.Protocol(b:sensor) <i>Input:</i> the receiving sensor; <i>Output:</i> $k_{a,b}^C$ and the set of co-operating sensors C</p> <ol style="list-style-type: none"> 1. Generate set C; 2. Set time-out Δ; 3. $k_{a,b}^C = 0$; 4. for all $c \in C$ do begin 5. $k_{a,c} = \text{Direct_Protocol}(c)$; 6. $a \rightarrow c : \langle a, c, E_{k_{a,c}}(\text{rich_key} b) \rangle$ 7. end; 8. $C' = C$; 9. while ($C' \neq \emptyset$ and (not elapsed(Δ))) do begin 10. $a \leftarrow c : \langle c, a, E_{k_{c,a}}(\text{HMAC}(ID_a, k_{c,b})) \rangle$; 11. $s = E_{k_{a,c}}^{-1}(E_{k_{a,c}}(\text{HMAC}(ID_a, k_{c,b})))$; 12. $C' = C' - \{c\}$; 13. $k_{a,b}^C = k_{a,b}^C \oplus s$ 14. end 15. $k_{a,b}^C = k_{a,b} \oplus k_{a,b}^C$; 16. $a \rightarrow b : \langle a, b, C E_{k_{a,b}^C}(\mathcal{H}(k_{a,b}^C)) \rangle$;
--

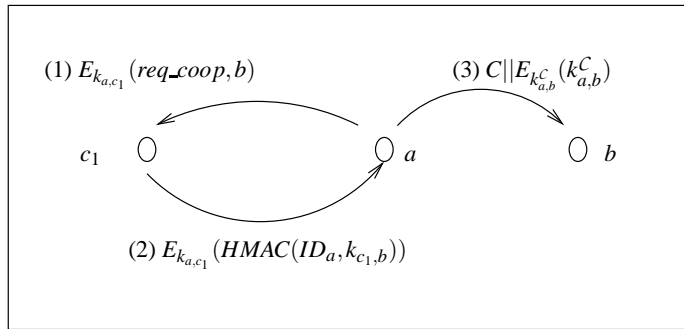


Figure 3: Example of co-operative approach with one co-operating sensor

- sensor failure resistance: if a sensor in C is not available for any reason (for instance, the sensor is destroyed or its battery exhausted), the protocol will not fail or deadlock; moreover, if some sensors do not answer to the request of co-operation within a certain time-out, sensor a can choose to add other sensors to C , in order to achieve a satisfactory level of security.
- no information leakage: since co-operating sensors provide a with a transformed, not-invertible image of their channel with b , no information on the effective channels $k_{c_i,b}$ is revealed;
- adaptiveness: if no information is available on the set of corrupted sensors but an upper bound on ω , set C can be chosen in such a way to secure all channels with the desired probability;
- load balance: the work-load generated by the Co-operating Protocol is equally distributed among all the sensors in C , one message each. Only sensor a has to send $m + 1$ messages (one for each of the co-operating sensors in C and one to b). The total cost for the WSN is thus $2m + 1$ potentially multi-hop messages. However, this cost is incurred only once, during the channel set up. Finally, note that sensor b does not need to send any message to build up the co-operative channel.

5.2 Security Analysis

In this section we analyze the probability of event $C_{a,b}^C \equiv$ “the co-operative channel between a and b , with set of co-operators $C = \{c_1, \dots, c_m\}$, is corrupted”. Recall that $C_{a,b}$ is the event “the direct channel between sensors a and b is corrupted”. Hence,

$$C_{a,b}^C \equiv C_{a,b} \wedge [\forall c \in C (C_{a,c} \vee C_{c,b})], \quad (6)$$

that is, coalition \mathcal{W} is able to corrupt the co-operative channel if and only if \mathcal{W} is able to corrupt the direct channel between a and b and, for each co-operating sensor c in C , either the channel between a and c , or the channel between b and c , or both. Note that, in the following, we explicitly considers the possible presence of corrupted sensors among the set of co-operating sensors.

An exact analysis of $\Pr[C_{a,b}^C]$ is complex. Indeed, even though it may appear counter-intuitive, events $C_{a,c'}$ and $C_{a,c''}$ are *not* independent, in spite of the independence of the key sets of c' and c'' . Similarly dependent are events $C_{a,c}$ and $C_{c,b}$. A way round this problem is to compute $\hat{\Pr}[C_{a,b}^C]$, the probability of event $C_{a,b}^C$ under the assumption that the keys assigned to sensors a , b and to the co-operating sensors in C are drawn from the pool *with replacement*. In other words, we assume that once key v_a^i is drawn from the pool and assigned to a , this key is replaced in the pool before key v_a^{i+1} is extracted. Conversely, the keys assigned to the corrupted sensors are drawn from the pool *without replacement*, as normally assumed in our protocol. Note that $\hat{\Pr}[C_{a,b}^C]$ is a lower bound on $\Pr[C_{a,b}^C]$. Indeed, key assignment from the pool with replacement possibly reduces the number of different keys a sensor holds due to possible multiple copies of the same key assigned to the same sensor. As a consequence, the number of keys shared by a , b , and the sensors in C , the “honest” sensors, is possibly reduced. Hence:

$$\Pr[C_{a,b}^C] = 1 - \Pr[\bar{C}_{a,b}^C] \leq 1 - \hat{\Pr}[\bar{C}_{a,b}^C]. \quad (7)$$

The following equation makes explicit the possible presence of corrupted sensors in C .

$$\begin{aligned}\hat{\Pr}[\bar{C}_{a,b}^C] &= \sum_{r=M}^m \hat{\Pr}[r = |C \setminus W|] \hat{\Pr}[\bar{C}_{a,b}^C | r = |C \setminus W|] = \\ &= \sum_{r=M}^m \frac{\binom{n-w}{r} \binom{w}{m-r}}{\binom{n}{m}} \hat{\Pr}[\bar{C}_{a,b}^C | r = |C \setminus W|],\end{aligned}\quad (8)$$

where $M = \max\{1, m - w\}$, and r is the number of non-corrupted sensors in C . Then,

$$\begin{aligned}\hat{\Pr}[\bar{C}_{a,b}^C | r = \#\{C \cap W\}] &= \\ &= \hat{\Pr}\left[\bigvee_{h=1}^r \left(\bigvee_{i=1}^k (v_a^i \in V_{c_h} \wedge v_a^i \notin W) \wedge \right.\right. \\ &\quad \left.\left. \bigwedge_{j=1}^k (v_b^j \in V_{c_h} \wedge v_b^j \notin W)\right)\right],\end{aligned}$$

where, without loss of generality, we assume that c_1, \dots, c_r are the r non-corrupted sensors in C . The above equation can be re-written as:

$$\begin{aligned}\hat{\Pr}\left[\bigvee_{h,i,j} \left((v_a^i \in V_{c_h} \wedge v_a^i \notin W) \wedge (v_b^j \in V_{c_h} \wedge v_b^j \notin W)\right)\right] &= \\ &= \sum_{s=1}^{rk^2} (-1)^{s+1} S_s.\end{aligned}$$

The generic term S_j is equal to:

$$\begin{aligned}S_j &= \binom{rk^2}{j} \left[\hat{\Pr}\left[\left(v_a^i \in V_{c_h} \wedge v_a^i \notin W\right) \wedge \right.\right. \\ &\quad \left.\left. \wedge \left(v_b^j \in V_{c_h} \wedge v_b^j \notin W\right)\right]\right]^j = \\ &= \binom{rk^2}{j} \left[\hat{\Pr}[v_a^i \in V_{c_h}] \cdot \hat{\Pr}[v_a^i \notin W] \cdot \right. \\ &\quad \left. \hat{\Pr}[v_b^j \in V_{c_h}] \cdot \hat{\Pr}[v_b^j \notin W] \right]^j = \\ &= \binom{rk^2}{j} \left[(1 - (1 - 1/P)^k)^2 (1 - k/P)^{2k\omega} \right]^j.\end{aligned}$$

Note in the above equation that, thanks to our assumption of key assignment to honest sensors with replacement, events $\{v_a^i \in V_{c_h}\}$, $\{v_a^i \notin W\}$, $\{v_b^j \in V_{c_h}\}$, and $\{v_b^j \notin W\}$ are now independent. Summarizing:

$$\begin{aligned}\hat{\Pr}[\bar{C}_{a,b}^C | r = \#\{C \cap W\}] &= \\ &= \sum_{i=1}^{rk^2} (-1)^{i+1} \binom{rk^2}{i} \left[(1 - (1 - 1/P)^k) (1 - k/P)^{k\omega} \right]^{2i}.\end{aligned}\quad (9)$$

Finally, our lower bound on $\Pr[C_{a,b}^C]$ is obtained by combining Equations 7, 8, and 9. The lower bound proves to be very tight when compared with the experimental results of Section 7.

5.3 DoS Attacks of Malicious Cooperators

In general, it is possible that corrupted sensors are chosen as cooperators in the Co-operative Protocol. This is explicitly considered in the analysis of the previous section and in the experiments, which

shows that channel confidentiality can be probabilistically guaranteed. However, it is interesting to explore all possible attacks from a cooperating sensor against other properties of the protocol.

Assume $\omega_i \in C \cap W$ is a corrupted sensor included into the set of cooperators for the channel set up from a to b . When ω_i is asked by a to provide its share, ω_i can behave as follows:

1. Sensor ω_i does not respond;
2. sensor ω_i sends a correct key $k_{\omega_i,b}$;
3. sensor ω_i sends a bogus key $\tilde{k}_{\omega_i,b}$.

Case 1 is equivalent to the case when a co-operating sensor is not available anymore, for instance it has been destroyed or its battery has run out. After the time-out Δ defined in Table 3, sensor a can decide to involve other sensors if the number of cooperating sensors is too low to guarantee its security requirements. Case 2 is the best choice if the attacker aims at breaking the channel confidentiality. Indeed, the channel is set up, and the attacker may have enough keys to recover $k_{a,b}^C$. Since this paper focuses on confidentiality, in both the analysis and the experimental evaluation we assume this is the default behavior of corrupted sensors. Note that the presence of malicious cooperators does not imply that the channel is corrupted.

Case 3 may result in a Denial of Service (DoS) on sensor a . Indeed, the channel built by sensor a with a bogus share does not match the channel locally computed by sensor b . Even though the goal of this paper is to address confidentiality, in the following we sketch a few countermeasures that can be taken to mitigate this sort of DoS attack. A first countermeasure is to randomly select another set C of co-operating sensors and to re-apply the co-operative protocol. In order to be an effective solution, set C should be chosen small enough to have $\Pr[C \cap W = \emptyset] > 1/t$, for some small positive integer t . In this way, after t iterations on the average, a good set of cooperators is found. Note that other subsets of cooperators can be added later to strengthen the same final cooperative channel. A second countermeasure is possible if a trusted channel to the center is available. Each sensor can store an individual secret key shared with the center, which also knows all the secret keys of the pool. When sensor a finds that a channel key $\tilde{k}_{a,b}^C$ could contain a bogus share, sensor a sends it to the center along with all the shares it used to build the key. Assuming that a is not corrupted, the center has all the information to track down efficiently the cheating sensors among C and b . Then, this is sent back to a . Note that the center is used only when a cheater is detected, and corrupted sensors are discouraged to give bogus shares in this setting. Finally, even when a channel to the center is not available, the presence of a bogus share gives the important information that the WSN is under attack.

6. PROBABILISTIC NODE TO NODE AUTHENTICATION

Node to node authentication is a key property. It is crucial to support a number of security functionalities. Any scheme for revocation of misbehaving nodes has its basis on the certainty that nodes identities are correctly detected, for example.

Assume sensor a successfully sets up a channel with sensor b by using the Direct Protocol. To what extent is sensor a sure of the identity of sensor b , and vice-versa? In principle, sensor identities could be faked since multiple copies of the same keys are distributed in the network. However, a fundamental property of the

pseudo-random key assignment, which is not a property of the random key assignment in [6], is that the indexes of the keys of any sensor are publicly known. This property supports the following authentication method: sensor b proves its identity by proving to know the keys sensor b is supposed to know.

In our Direct Protocol, any set \mathcal{W} of colluding sensors that pretend to be sensor a during a channel set up with b must have all the keys in $V_a \cap V_b$ to succeed. Hence, the coalition can fake the identity of a only if $V_a \cap V_b \subseteq W$, since \mathcal{W} has to locally compute a key depending on all keys in $V_a \cap V_b$. We already know that this is possible with probability $\Pr[C_{a,b}]$, which is the probability that the channel confidentiality is compromised. Consequently, as far as the system is built in such a way to guarantee high confidentiality of node to node channels, it also implicitly guarantees node to node authentication with the same probability, at no additional cost. It is easy to realize that the Cooperative Protocol also has the same property. Finally, note that the above results do not depend on the network size n . Hence, our proposed node to node probabilistic authentication has excellent scaling properties.

7. EXPERIMENTAL EVALUATION AND DISCUSSION

For any given choice of parameters P , k , ω , m , and network size n , we performed our simulations iterating the following steps:

1. a pool composed of P random keys is generated;
2. a collection \mathcal{N} of n sensors, each with a distinct ID, is generated;
3. sets C of m sensors and \mathcal{W} of ω sensors are randomly, uniformly, and independently extracted from \mathcal{N} ; note that it is possible that $C \cap \mathcal{W} \neq \emptyset$;
4. for each sensor $c \in C$, we uniformly extract from the pool a set V_c of k keys; the same is independently done for communicating sensors a , b , and for each sensor in \mathcal{W} .
5. for this particular assignment of keys to sensors, we compute:
 - (a) $V_a \cap V_b \neq \emptyset$, whether the direct channel between a and b exists;
 - (b) $V_a \cap V_b \subseteq W$, whether the direct channel between a and b , if it exists, is compromised;
 - (c) for all $c \in C$ $V_a \cap V_c \neq \emptyset$ and $V_c \cap V_b \neq \emptyset$, whether the direct channels between a and c and between c and b exist;
 - (d) for all $c \in C$ $V_a \cap V_c \subseteq W$ $V_c \cap V_b \subseteq W$, whether the direct channel between a and c and between c and b , if they exist, are compromised;
 - (e) whether channel key $k_{a,b}^C$ is compromised.

We computed each of our final results from 10^5 runs of the above steps.

7.1 Simulation Results

In Figure 4 we simulated the behavior of the Co-operative Protocol when $|C| = 4, 6, 8$, $\omega = 8$, pool size $P = 1000$, and the number of keys per sensor k varies from 40 to 200. Here, a small number of co-operating sensors are sufficient to increase the resilience to

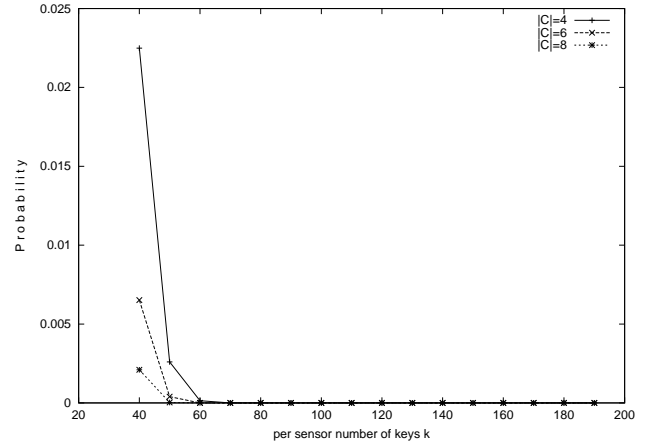


Figure 4: Channel corruption probability for different number of co-operating sensors, $P=1000$ and $\omega = 8$.

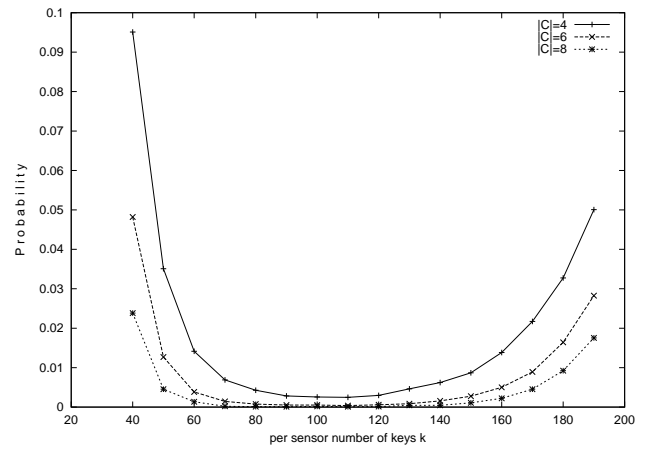


Figure 5: Channel corruption probability for different number of co-operating sensors, $P=1000$ and $\omega = 16$.

corruption even for small values of k . In Figure 5, where $\omega = 16$, four co-operating sensors are not sufficient to reduce the channel corruption probability to a negligible value. However, slightly increasing the set of co-operating sensors produces a big reduction of corruption probability. In Figure 6 we assume $\omega = 32$ and the pool is composed of ten thousand keys ($P = 10000$). For such a number of corrupted sensors, the experiments show that it is possible to reduce the corruption probability to negligible values by choosing an appropriate trade off, e. g. $|C| = 8$ and $k = 160$.

Finally, note that the results for both the Direct and the Co-operative Protocols are independent from the network size, hence they show excellent scalability.

8. CONCLUSION

This paper has described a probabilistic model and two protocols to establish secure pair-wise communication channel between any pair of sensors in a WSN. The proposed protocols shows many advantages. The number of keys to be assigned to each sensor is bounded by a constant, regardless of size of the WSN, given the number of corrupted sensors in the WSN and the desired level of security. The cooperative protocol is adaptive with respect to the

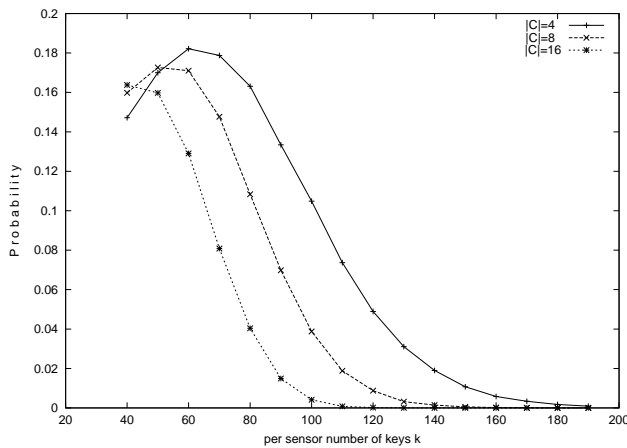


Figure 6: Channel corruption probability for different number of co-operating sensors, $P=10000$ and $\omega = 32$.

security threats, that is, should the number of corrupted sensors in the WSN increase, it is possible to achieve the desired level of security at the price of an increase in communications. Note that this overhead is paid only during the channel set up phase, while no additional overhead is incurred for the subsequent pair-wise communications. The above considerations are supported by both probabilistic analysis and extensive simulations.

Our choice of using pseudo-random generation of the key indexes of each key ring using the sensor's id as the seed has advantages and disadvantages. This way of deriving key rings reduces computational and communication overhead compared with the generation of the key indexes using a secret seed [6, 10]. Also, our approach seems to provide a few other important security benefits. First, as seen in Section 6, implicit and probabilistic mutual authentication between sensors can be supported with no overhead. Next, an attacker cannot add to the WSN new sensors with different identifiers by replicating the same stolen key ring. Indeed, every key ring can be used (with high probability) by only one sensor id - malicious sensors with different identifiers must be loaded with different key rings. Moreover, sensors do not have to respond to key discovery requests from unknown devices, like in [6] and [10], simply because there is no key discovery. This greatly mitigates a number of efficient DoS attacks.

On the other hand, since key rings are derived from the sensor id, the attacker could select a smaller set of sensors that will give him a large subset of the main key pool. However, it is not straightforward to avoid related problems even using a secret seed and a key discovery phase. With a key discovery phase approach, a malicious eavesdropper, based on the information exchanged between legitimate sensors, can identify which sensors hold the maximum number of keys yet to be compromised, and, applying a greedy technique, plan its attacks against them. For example, the attacker could try to open the messages exchanged during the key discovery phase between legitimate sensors a and b . If all the already compromised keys do not work, then the messages are related to uncompromised keys, and a and b are good candidates for the next attack.

Based on the pros and cons discussed above, the pseudo-random generation of key rings using the sensor's id seems a reasonable

trade-off between security and performance.

Among future works, we intend to investigate whether it is possible to enhance further the resilience of the co-operative channel to the set of corrupted sensors, while reducing the number of messages required by the protocol.

9. REFERENCES

- [1] Giuseppe Ateniese, Michael Steiner, and Gene Tsudik. New multiparty authentication services and key agreement protocols. *IEEE Journal on Selected Areas in Communications*, 18(4), 2000.
- [2] S. Basagni. Distributed clustering for *ad hoc* networks. In *Proceedings of the 1999 International Symposium on Parallel architecture, Algorithms, and Networks (I-SPAN'99)*, IEEE computer Society, pages 310–315, 1999.
- [3] Stefano Basagni, Kris Herrin, Danilo Bruschi, and Emilia Rosti. Secure pebblenets. In *Proceedings of the 2001 ACM International Symposium on Mobile ad hoc networking & computing*, pages 156–163. ACM Press, 2001.
- [4] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. Keying hash functions for message authentication. In Neal Koblitz, editor, *Advances in Cryptology – Crypto 96 Proceedings*, volume 1109 of *Lecture Notes in Computer Science*. Springer-Verlag, 1996.
- [5] D. W. Carman, P. S. Kruus, and B. J. Matt. Constraints and approaches for distributed sensor network security. Technical Report #00-010, NAI Labs, 2000.
- [6] Haowen Chan, Adrian Perrig, and Dawn Song. Random Key Predistribution Schemes for Sensor Networks. In *Proceedings of the IEEE Symposium on Security and Privacy*, Oakland, California-USA, pages 197–213, 2003.
- [7] Jyh-Cheng Chen, Krishna M. Sivalingam, and Prathima Agrawal. Performance comparison of battery power consumption in wireless multiple access protocols. *Wireless Networks*, 5(6):445–460, 1999.
- [8] Roberto Di Pietro, Luigi V. Mancini, and Sushil Jajodia. Secure selective exclusion in ad hoc wireless network. In *Proceedings of the eighteenth IFIP International Information Security Conference*, pages 423–434. Kluwer, May 5-7, 2002.
- [9] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, 1976.
- [10] Laurent Eschenauer and Virgil D. Gligor. A key-management scheme for distributed sensor networks. In *Proceedings of the 9th Computer Communication Security (CCS 2002)*, November 2002, Washington, MA.
- [11] A.D. Amis et al. Max-min d-cluster formation in wireless *ad hoc* networks. In *Proceedings of INFOCOM 2000*, pages 32–41, 2000.
- [12] John Haigh. *Probability Models*. Springer, 2002. ISBN 1-85233-431-2.

- [13] Jason Hill, Robert Szewczyk, Alec Woo, Seth Hollar, David Culler, and Kristofer Pister. System architecture directions for networked sensors. *ACM SIGPLAN Notices*, 35(11):93–104, 2000.
- [14] Yih-Chun Hu, Adrian Perrig, and David B. Johnson. Ariadne: a secure on-demand routing protocol for ad hoc networks. In *Proceedings of the eighth annual international conference on Mobile computing and networking*, pages 12–23. ACM Press, 2002.
- [15] J.M. Kahn, R.H. Katz, and K.J. Pister. Mobile networking for smart dust. In *Proceedings of the ACM/IEEE Intl. Conf. on Mobile Computing and Networking (MobiCom99)*, pages 271–278. ACM Press, 1999.
- [16] S. Marti and K.L.M. Baker. Mitigating routing misbehavior in mobile *ad hoc* networks. In *Proceedings of the sixth annual international conference on Mobile computing and networking*, pages 255–265, 2000.
- [17] NIST. Fips pub 140-1, security requirements for cryptographic modules. National Institute of Standard and Technology, January 1994.
- [18] Charles E. Perkins. *Ad Hoc Networking*. Addison Wesley Professional, 2001. ISBN 0-201-30976-9.
- [19] Adrian Perrig, Robert Szewczyk, Victor Wen, David Culler, and J. D. Tygar. Spins: security protocols for sensor networks. In *Proceedings of the seventh annual international conference on Mobile computing and networking*, pages 189–199. ACM Press, 2001.
- [20] Lakshmi Ramachandran, Manika Kapoor, Abhinanda Sarkar, and Alok Aggarwal. Clustering algorithms for wireless ad hoc networks. In *Proceedings of the 4th international workshop on Discrete algorithms and methods for mobile computing and communications*, pages 54–63. ACM Press, 2000.
- [21] E.M. Royer and C-K Toh. A review of current routing protocols for *ad hoc* mobile wireless network. In *IEEE Personal Communication Magazine*, pages 46–55, April 1999.
- [22] A. Sinha and A. Chandrakasan. Dynamic power management in wireless sensor networks. In *IEEE Design and Test of Computer*, March/April 2001.
- [23] M. Steiner, G. Tsudik, and M. Waidner. Key agreement in dynamic peer groups. *IEEE Transactions on Parallel and Distributed Systems*, 11(8):769–780, 2000.
- [24] Yongguan Zhang and Wenke Lee. Intrusion detection in wireless *ad hoc* networks. In *Proceedings of the sixth annual international conference on Mobile computing and networking*, pages 275–283, 2000.
- [25] Sencun Zhu, Sanjeev Setia, and Sushil Jajodia. A distributed group key management protocol for ad hoc networks. Unpublished manuscript, December 2002, George Mason University, VA-USA.