SecLEACH – A Random Key Distribution Solution for Securing Clustered Sensor Networks

Leonardo B. Oliveira[1], **Hao Chi Wong**[2], **Marshall Bern**[2]
Eduardo Habib [3], **Antonio A. F. Loureiro**[3], **Ricardo Dahab**[1]

[1] Instituto de Computação
Universidade Estadual de Campinas (UNICAMP)
Campinas, SP

[2]Palo Alto Research Center (PARC)
Palo Alto, CA, EUA

[3]Departamento de Ciência da Computação
Universidade Federal de Minas Gerais (UFMG)
Belo Horizonte, MG

{leob,rdahab}@ic.unicamp.br,{hcwong,bern}@parc.com,{habib,loureiro}@dcc.ufmg.br,

Abstract. *Wireless sensor networks are ad hoc networks comprised mainly of small sensor nodes with limited resources, and can be used to monitor areas of interest. Recent work has shown that clustered sensor networks can increase system throughput, decrease system delay, and save energy. In this paper, we show that random key predistribution, first proposed and studied in the context of flat wireless sensor networks, is a very attractive key distribution solution for clustered sensor networks with rotating cluster heads, such as LEACH.*

1. Introduction

Wireless sensor networks (WSNs) [1, 2] are ad hoc networks comprised mainly of small sensor nodes with limited resources (low power, low bandwidth, and low computational and storage capabilities) and one or more base stations (BSs), which are much more powerful nodes that connect the sensor nodes to the rest of the world. WSNs are rapidly emerging as a technology for large-scale, low-cost, automated sensing and monitoring of different environments of interest. Potential WSN applications range from battlefield reconnaissance to environmental protection.

When embedded in critical applications, WSNs are likely to be attacked. And just like any other type of networks, they are vulnerable [3, 4]. Aside from the well known vulnerabilities due to wireless communication and *ad-hoc*ness, WSNs face other security-related challenges, including lack of physical protection (sensor nodes are low-cost devices that are unlikely to be made tamper-resistant or tamper-proof) and their deployment in open, unattended, and often hostile environments (which makes them easily accessible to random individuals and malicious parties alike). It is therefore crucial to add security to these networks, specially those that are part of mission-critical applications.

One of the main issues in securing networks through cryptographic methods is key distribution, which has been intensively studied recently [5–19] in the context of WSNs. Besides security, efficiency is a factor that should be highlighted when evaluating a key distribution scheme for WSNs, as they are so highly resource-constrained (e.g, Mica2 Motes feature a 7.8 MHz processor and 4 KB of RAM memory [20, 21])

Also, a large number of network architectures have been propose d [22–24] for WSNs, and a key distribution solution that is efficient for one architecture is likely not to be the best for another, as different network architectures exhibit different communication patterns.

Cluster-based organization (e.g., [25–29]) has been proposed for ad hoc networks in general and sensor networks in particular. In cluster-based networks, nodes are typically organized into clusters, with cluster heads (CHs) relaying messages from ordinary nodes in the cluster to the BSs. Clustered WSNs were first proposed for various reasons including scalability and energy efficiency. Those with rotating CHs, like LEACH (Low Energy Adaptive Clustering Hierarchy) [25], are also interesting in terms of security, as their routers (the CHs), which are more prominent targets for adversaries because of their role in routing, rotate from one node to another periodically. This rotation helps in security, because it makes it harder for an adversary to identify the routing elements and compromise them [3].

Adding security to LEACH-like protocols is challenging. Resource constraints in sensor nodes make more conventional solutions (e.g., those based on public key methods) inapplicable. Furthermore, its dynamic and periodic rearranging the network's clustering makes it necessary to provide keys to dynamically and periodically changing links, and difficult for us to rely on long-lasting node-to-node trust relationships to make the protocol secure.

In this paper, we focus on providing security to pairwise communications in LEACH-like protocols. To this end, we first propose SecLEACH, a modified version of LEACH that bootstraps its security from random key predistribution. We then give a detailed analysis and performance evaluation of our scheme, and show how it is impacted by the various parameters. Our main contributions in this paper are: 1) to have provided an efficient solution for securing pairwise communications in LEACH, and 2) to have evaluated application of random key predistribution to dynamic cluster-based sensor network protocols.

The rest of this paper is organized as follows. In section 2, we introduce the original LEACH protocol, and discuss its vulnerabilities. In section 3, We discuss what is needed to cryptographically secure its communications, as well as existing solutions. We present our solution (SecLEACH) in section 4, and analyze its performance in section 5. Finally, we discuss related work and conclude in sections 6 and 7, respectively.

2. LEACH and its vulnerabilities

LEACH [25] assumes two types of network nodes: a more powerful BS and a larger number of resource-scarce sensor nodes. In WSNs, resource-scarce nodes do not typically communicate directly with the BS for two reasons. One, these nodes typically have transmitters with limited transmission range, and are unable to reach the BS directly. Two, even if the BS is within a node's communication range, direct communication typically demands a much higher energy consumption.

A more energy efficient alternative takes advantage of one's neighboring nodes as routers. Nodes that are farther away send their messages to intermediate nodes, which then forward them towards the BS in a multi-hop fashion. The problem with this approach is that, even though peripheral nodes actually save energy, the intermediate nodes, which play the role of routers, spend additional energy receiving and transmitting messages, and end up having a shortened lifetime,

LEACH assumes every node can directly reach a BS by transmitting with sufficiently high power. However, to save energy and avoid the aforementioned problem, LEACH uses a novel type of routing that randomly rotates routing nodes among all nodes in the network. Briefly, LEACH works in rounds, and in each round, it uses a distributed algorithm to elect CHs and dynamically cluster the remaining nodes around the CHs. To avoid energy drainage of CHs, they do not remain CHs forever; nodes take turns in being CHs, and energy consumption spent on routing is thus distributed among all nodes.

Using a set of 100 randomly distributed nodes, (and a BS located at 75m from the closest node) simulation results [25] show that LEACH spends up to 8 times less energy than other protocols. To be fair, the energy saving comes from a number of sources other than just dynamic cluster-based communication: data fusion (CHs do data fusion before sending them to the BS), node sleeping (given that only CHs need to forward messages, the remaining nodes are activated

Setup phase

1. $H \Rightarrow \mathcal{G}$: id_H, crc, adv
2. $A_i \rightarrow H$: $id_{A_i}, id_H, crc, \mathsf{join\_req}$
3. $H \Rightarrow \mathcal{G}$: $id_H, (\ldots, \langle id_{A_i}, T_{A_i} \rangle, \ldots), crc, \mathsf{sched}$

Steady-state phase

4. $A_i \rightarrow H$: id_H, D_{A_i}, crc
5. $H \rightarrow BS$: $id_H, id_{BS}, \mathcal{F}(\ldots, D_{A_i}, \ldots), crc$

The various symbols denote:

A_i, H, BS :	An ordinary node, a cluster head, and the base station, respectively	adv, join_req, sched :	String identifiers for message types
$\mathcal{G}$:	The set of all nodes in the network	crc :	Cyclic Redundancy Check
$id_{A_i}, id_H, id_{BS}, id_k$:	Node A_i, H, BS, and k's ids, respectively	$\langle x, T_x \rangle$:	A node id x and its time slot T_x in its cluster's TDMA schedule
$\Rightarrow, \rightarrow$:	Broadcast and unicast transmissions, respectively	D_x :	Sensing report from node X
		$\mathcal{F}$:	Data fusion function

Figure 1: LEACH protocol

only when they themselves are transmitting, and remain in sleep mode for a reasonable amount of time), and transmitter calibration (nodes calibrate their transmitters in such a way that they are only high enough to reach the CH).

2.1. Protocol Description

Rounds in LEACH (Fig. 1) have predetermined duration, and have a *setup* phase and a *steady-state* phase. Through synchronized clocks, nodes know when each round starts and ends.

The setup consists of three steps. In the *advertisement* step (Step 1), nodes decide probabilistically whether or not to become a CH for the current round (based on its remaining energy and a globally known desired percentage of CHs). Those that will broadcast a message (adv) advertising this fact, at a level that can be heard by everyone in the network. To avoid collision, the CSMA-MAC protocol is used. In the *cluster joining* step (Step 2), the remaining nodes pick a cluster to join based on the largest received signal strength of a adv message, and communicate their intention to join by sending a join_req (join request) message using CSMA-MAC. Once the CHs receive all the join requests, the *confirmation* step (Step 3) starts with the CHs broadcasting a confirmation message that includes a time slot schedule to be used by their cluster members for communication during the steady-state phase. Given that the CHs' transmitters and receivers are calibrated, balanced and geographically distributed clusters should result.

Once the clusters are set up, the network moves on to the steady-state phase, where actual communication between sensor nodes and the BS takes place. Each node knows when it is its turn to transmit (Step 4), according to the time slot schedule. The CHs collect messages from all their cluster members, aggregate these data, and send the result to the BS (Step 5). The steady-state phase consists of multiple reporting cycles, and lasts much longer compared to the setup phase.

2.2. Security vulnerabilities

Like most routing protocols for WSNs, LEACH is vulnerable to a number of security attacks [3], including jamming, spoofing, replay, etc. However, because it is a cluster-based protocol, relying fundamentally on the CHs for routing, attacks involving CHs are the most damaging. If an intruder manages to become a CH, it can stage attacks such as sinkhole and selective forwarding, thus disrupting the workings of the network. Of course, the intruder may leave the routing alone, and try to inject bogus sensor data into the network, one way or another. A third type of attack is (passive) eavesdropping.

It is worth noting that LEACH is more robust against insider attacks than most other routing protocols [3]. In constrast to more conventional multihop schemes where nodes around the BS are especially attractive for compromise (because they concentrate all network-to-BS communication flows), LEACH CHs communicate directly with the BS, can be anywhere in the network, and change from round to round. All these characteristics make it harder for an adversary to identify and compromise strategically more important nodes.

3. Adding Security to LEACH: Background

One of the first steps to be taken to secure a WSN is to prevent illegitimate nodes from participating in the network. This access control can preserve much of a network's operations, unless legitimate nodes have been compromised. (Note that access control does not solve all security problems in WSNs. E.g., it is ineffective against DoS attacks based on jamming wireless channels, or manipulation of a node's surrounding environment to induce the reporting of fabricated conditions.)

Access control in networks has typically been implemented using cryptographic mechanisms, which rely critically on key distribution. There are a number of standard key distribution schemes in the security literature [30], most of which are ill-suited to WSNs: public key based distribution, because of its processing requirements; global keying, because of its security vulnerabilities; complete pairwise keying, because of its memory requirements; and those based on a key distribution center, because of its inefficiency and energy consumption [8].

Some key distribution schemes [6–8, 10, 12, 16, 17, 19] have been specifically designed for WSNs, but they are designed with particular network organizations in mind, and are inadequate to others. For instance, LEAP [8], a recently proposed scheme based on local distribution of keys among nodes in a neighborhood, is rather efficient for flat networks where the nodes have a limited neighborhood, but is inadequate for protocols like LEACH, where a node's neighborhood is in fact the whole network. In what follows, we discuss the network model assumed in LEACH, and the requirements it sets for key distribution.

3.1. Key Distribution for LEACH: Requirements and Constraints

Our discussion in section 2.2 shows the need for the nodes to authenticate each other as legitimate members of the network both in the setup interactions and the sensor data reporting communications. Given the communication patterns in LEACH, two different types of authentication are required: authenticated broadcast, for broadcasts from the CHs to the rest of the network (Fig. 1, steps 1 and 3); and the basic pairwise authentication for the remaining (node-to-CH and CH-to-BS) communications.

Symmetric-key authenticated broadcasts, both global (μTESLA [31]) and local (LEAP [8]), have been proposed for WSNs, and share the core idea of using a one-way key chain to achieve authentication. These schemes cannot be applied, *as is*, to LEACH for two reasons: 1) the key chain would require a significant storage space from the broadcasting CHs, and more importantly, 2) all nodes in the network would need to store one key for each node in the network, which is neither practical nor scalable. (Note that all nodes need to store one key for each node in the network because an ordinary node needs to be able to authenticate the CHs in each round, which can be arbitrary nodes in the network.)

The basic pairwise authentication is also challenging to implement in LEACH, because of key distribution issues. Given that an ordinary node needs to be ready to join any CH, which could be any node in the network, it needs to have shared pairwise keys with all nodes in the network. Just like in authenticated broadcast, this is neither practical, nor scalable.

Cryptographic protection for LEACH has been studied before. In [32], Ferreira *et al* propose SLEACH (Fig. 2), based on two symmetric keys for each node: a pairwise key shared with the BS; and a commitment to a key chain held by the BS, which is used in its authenticated broadcast. SLEACH divides authenticated broadcasts by the CHs into two smaller steps, leveraging

Setup phase

$$
\begin{array}{lrl}
1.1. & H \Rightarrow \mathcal{G}: & id_H, \mathsf{mac}_{k_H}(id_H \mid c_H \mid \mathsf{adv}) \\
 & A_i: & \text{store}(id_H) \\
 & BS: & \text{if } \mathsf{mac}_{k_H}(id_H \mid c_H \mid \mathsf{adv}) \text{ is valid,} \\
 & & \text{add}(id_H, V) \\
1.2. & BS \Rightarrow \mathcal{G}: & V, \mathsf{mac}_{k^j}(V) \\
1.3. & BS \Rightarrow \mathcal{G}: & k^j \\
 & A_i: & \text{if } \left(f\left(k^j\right) = k^{j+1}\right) \text{ and } \left(id_H \in V\right), \\
 & & id_H \text{ is authentic} \\
2. & A_i \rightarrow H: & id_{A_i}, id_H, crc, \mathsf{join\_req} \\
3. & H \Rightarrow \mathcal{G}: & id_H, (\ldots, \langle id_{A_i}, T_{A_i}\rangle, \ldots), crc, \mathsf{sched}
\end{array}
$$

Symbols as previously defined, with the following additions:

$\mathsf{mac}_k()$:	MAC calculated using k	store(id_h) :	Store id h for future validation
k_x :	Symmetric key shared by A_i and BS	add(id_h, V) :	Add id h to V
k^j :	Keys in a one-way key chain;	V :	An array of node ids
		$f()$:	One-way hash function

Figure 2: SLEACH's setup protocol

on the BS, who is trusted and has more resources. Briefly, each CH sends (Step 1.1) a slightly modified adv message consisting of: 1) the id of the CH in plaintext, used by the ordinary nodes as before; and 2) a MAC produced using the key the CH shares with the BS (which will be used by the BS for the purpose of authentication). The BS waits to hear and authenticate (modified) adv msgs from all CHs; compiles the list of legitimate CHs; and sends the list to the network using the μTESLA broadcast authentication scheme (Steps 1.2 and 1.3). Ordinary nodes now know which of the advs they received are from legitimate nodes, and can proceed with the rest of the original protocol, choosing the CH from the list broadcast by the BS. We reproduce SLEACH's setup protocol in Fig. 2; for clarity of presentation, we leave join_req and sched msgs unmodified.

Using only two keys per node, SLEACH does not manage to provide a complete and efficient solution for node-to-CH authentication. In particular, join_req msgs in the setup protocol are not authenticated; and sensing reports from the ordinary nodes (in the steady-state phase) are encrypted using pairwise keys shared between the nodes and the BS. This means that the CHs are prevented from reading the sensing reports and carrying out data aggregation; in fact, they forward all the reports, which incurs a large energy consumption.

In this paper, we address the key distribution problem for node-to-CH authentication in LEACH.

4. SecLEACH – Applying Random Key Distribution to LEACH

In this section, we first briefly describe random key predistribution (section 4.1), then show how it can be incorporated to LEACH to enable node-to-node authentication (section 4.2).

4.1. Random Key Predistribution Schemes

Random key predistribution for WSNs was first proposed by Eschenauer and Gligor [6], and has since been studied by several research groups [7, 9, 12, 14, 16, 17]. In a random key predistribution scheme, each node is assigned a set of keys drawn from a much larger key pool. Different schemes have different assignment algorithms, but they all result in probabilistic key sharing among the nodes in the network. For securing communication, those that do share keys can set up a secure link between them. These links can then be used to bootstrap other secure links.

Setup phase

1. $H \Rightarrow \mathcal{G}:$ $\quad id_H, seed_H, nonce, crc, \mathsf{adv}$
2. $A_i \rightarrow H:$ $\quad id_{A_i}, id_H, id_K, \mathsf{join\_req}, \mathsf{mac}_{k[r]}(id_{A_i} \mid id_H \mid id_K \mid nonce \mid \mathsf{join\_req}),\ id_K \subset \mathcal{R}_H$
3. $H \Rightarrow \mathcal{G}:$ $\quad id_H, (\ldots, \langle id_{A_i}, T_{A_i} \rangle, \ldots), \mathsf{sched}$

Steady-state phase

4. $A_i \rightarrow H:$ $\quad id_H, D_{A_i}, \mathsf{mac}_{k_{[r]}}(id_H \mid D_{A_i} \mid nonce + 1)$
5. $H \rightarrow BS:$ $\quad id_H, id_{BS}, \mathcal{F}(\ldots, D_{A_i}, \ldots), \mathsf{mac}_H(id_H \mid id_{BS} \mid \mathcal{F}(\ldots, D_{A_i}, \ldots) \mid c_H)$

Symbols as previously defined, with the following additions:

$\mathcal{R}_H$:	The set of key ids of H	c_x :	Counter shared by X and BS
$k_{[r]}$:	Random key shared by X and H	$seed_x$:	PRNG seed that generates key ids of X

Figure 3: SecLEACH protocol

To bootstrap security using Eschenauer and Gligor's original scheme [6], a network goes through three phases: *key predistribution*, *shared-key discovery*, and *path-key establishment*. During key predistribution phase, which takes place prior to network deployment, a large pool of S keys and their ids are generated. Each node is then assigned a ring of m keys, drawn from the pool at random, without replacement. In the shared-key discovery phase, which takes place during network setup, all nodes broadcast the ids of the keys on their key rings. Through these broadcasts, a node will find out with which of their neighbors (as determined by communication range) they share a key. These keys can then be used for establishing secure links between the two neighbors. Finally, during path-key establishment, pairs of neighboring nodes that do not share a key can set up their own keys, as long as they are connected by two or more secure links at the end of shared key discovery.

Note that because of the way keys are assigned, a key can be found in more than two nodes, and used in multiple communication links. When a node is compromised, all its keys are compromised, and all the links secured by these keys are also compromised.

4.2. SecLEACH – Protocol Description

We propose to use random key predistribution to enable node-to-node authentication in LEACH. In a nutshell, prior to network deployment, a large pool of S keys and their ids are generated. Each node is then assigned a ring of m keys, drawn from the pool at random, without replacement. The ring is constructed by using a pseudorandom number generator (PRNG) function, i.e., for each node a given seed is used as input to the PRNG that then generate the key ids. The corresponding keys, the PRNG, and the seed are then loaded into the node. The LEACH clustering algorithm can then be run with the following modifications: when a self-elected CH broadcasts its adv msg , it includes a seed that, employed in conjunction with the PRNG, may be used to generate the ids of the keys in its key ring; the remaining nodes now cluster around the closest CH with whom they share a key. Fig. 3 shows the details of our SecLEACH protocol.

In Step 1, a self-elected CH H broadcasts its id id_H, the PRNG seed, and a nonce, for freshness. In Step2, ordinary nodes A_i choose the closest CH with whom they share a key, and send it a join_req msg. A MAC produced using the key (or one of the keys) they share and using the nonce from H's broadcast in Step 1 is included in the msg. The id id_K of the key is also included in the msg, so that the receiving CH knows which key to use to verify the MAC. In Step 3, the CHs send the time slot schedule to the nodes that chose to join their clusters, and conclude the setup phase.

In the steady-state phase, node-to-CH communications (Step 4) are protected using the same key used to protect the join_req msg in Step 2. A value computed from the nonce (shown as "$nonce + 1$") is also included in the MAC for freshness. The CHs can now verify the sensing

reports'MACs they receive, perform data aggregation, and send the aggregate result to the BS (Step 5). The aggregate result is along with a MAC produced using both the symmetric key and a counter (for freshness) shared between the CH and the BS.

Fig. 3 shows only one reporting cycle in the steady-state phase. In practice, there will be multiple cycles in a round. In each round, the value of the "freshness token" ("$nonce + 1$" in Step 4, and "c_H" in Step 5) needs to be incremented by 1. Note also that, in Fig. 3, Steps 1 and 3 are not protected. In fact, the random key predistribution is not helpful for authenticating these broadcasts. We envision our scheme to work in conjunction with the authenticated broadcast proposed in SLEACH (Fig. 2).

At the end of the clustering process, we expect that a fraction of the ordinary nodes will be matched with a CH, though not necessarily the one they would have matched with in the basic LEACH, because of key sharing constraints; the remaining would not have any CH to match with. We call these nodes orphans.

There are different ways to deal with the orphans: we can have them sleep for the round; we can add a small protocol that would allow the "already-adopted children" to bring the orphans into their clusters; or we can have them communicate directly with the BS for the round. In any case, we will show in Section 5 that the percentage of orphans will depend on the size of the key pool, the size of the key ring, and the *absolute* number of CHs, and will have a negligible impact on the performance of the network.

4.3. Security Analysis

SecLEACH provides authenticity, integrity, confidentiality, and freshness to node-to-node communications. The msg in Step 2, Fig. 3, makes use of a MAC produced from a key in the key pool; and a successful verification of this msg allows H to conclude that the msg originated from a legitimate node in the network. Because the msg's MAC includes the nonce from Step 1, H can also conclude that it is not a stale msg being replayed.

The same observations apply to the msg in Step 4. The freshness of all subsequent sensor reports from the ordinary nodes to their BS is guaranteed by nonces values that are incremented each time. For the msg in Step 5, the freshness is guaranteed by the counter value shared between the CH and the BS; the counter value also being incremented each time the CH sends a new report to the BS.

5. Evaluation of our scheme

Random key predistribution schemes have all been introduced and studied in the context of flat networks, which come with the following assumptions: 1) the nodes have antennas with limited transmission range; and 2) node-to-BS communications are multi-hop, with the nodes each relying on their neighbors to forward messages towards the BS. In this context, any two nodes within each other's transmission range has a communication link between them, and a forwarding route can be established between any two nodes (including the BS) as long as one can overlay a connected graph on the network using these range-defined links.

In flat networks where security is to be bootstrapped from random key predistribution, there is a (secure) link between two nodes only if they are within each other's communication range and share a key. In this new context, a (secure) forwarding route can be established between any two nodes (including the BS) only if one can overlay a connected graph on the network using secure links. Given that it is possible that a physical (range-defined) link will be (logically) severed by lack of a shared key between the two end nodes, one needs to choose the parameters S (size of the key pool) and m (size of the key ring) in such a way that the resulting network is still (securely) connected, with high probability.

In the context of LEACH, the assumptions are slightly different: 1) any node in the network is reachable from any other node in single hop; but 2) node-to-BS communications are

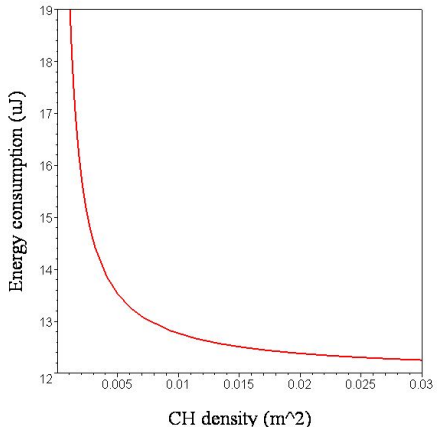

Figure 4: Average node-to-CH energy consumption

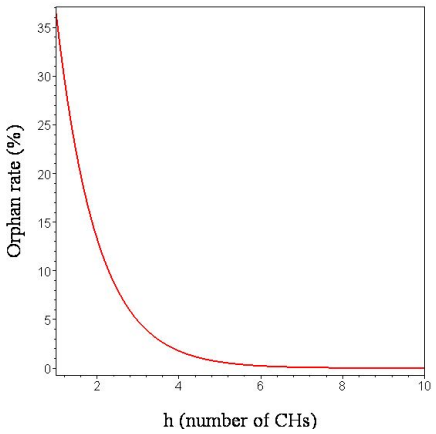

Figure 5: Orphan rate, for sl=0.99

typically carried out in two-hops: from ordinary nodes to CHs, and from CHs to the BS. Because of the first assumption, any ordinary nodes can theoretically join any CH; in practice, they choose the closest to save energy. For maximum energy efficiency, however, a network needs to use just the right number of CHs, as different number of CHs leads to different energy consumptions.

In SecLEACH, because of the constraints imposed by key sharing, not all CHs are accessible to all ordinary nodes. In fact, depending on the values of S and m, which determine the probability that two nodes will share a key, an ordinary node will have a larger or smaller number of CHs to choose from. To achieve maximum energy efficiency in the context of SecLEACH, therefore, one needs to find right values for S, m, and the number of CHs. In what follows, we show how different parameter values impact a network, in terms of security and energy efficiency.

5.1. Parameter Values and Their Impact on Performance

Given a WSN, the amount of storage reserved for keys in each node is likely to be a preset constraint, which makes the size of the key ring m a fixed parameter in the system. Once m is set, the choice of S will impact the system in two ways:

1. Its security level:
   Given a (S, m)-network, $\frac{m}{S}$ is the probability that a randomly chosen link will be compromised when a node that is not either end of the link is compromised. The *security level* sl of a (S, m)-network can then be defined as:

   $$sl = 1 - \frac{m}{S}$$

   which gives the probability that a randomly chosen link is not compromised when a node that is not either end of the link is compromised.
   Note that given a fixed m, the larger the S, the larger the *sl* (the higher the security level).
2. The probability that two nodes will share a key:
   Given any two nodes in a (S, m)-network, the probability P_s that they will share a key is given by:

   $$P_s = 1 - P_{\bar{s}}$$

   where $P_{\bar{s}}$, the probability that they will not share a key, is given by:

   $$P_{\bar{s}} = \frac{[(S-m)!]^2}{S!(S-2m)!}$$

   It has been shown that given a fixed m, the larger the S, the smaller the P_s [6].

The number h of CHs in the network is another parameter in the system. In LEACH, the density of CHs in a network determines the average distance between a node and its closest CH.

This distance, in turn, determines the amount of energy needed in node-to-CH communications: the denser the CHs, the shorter the average node-to-CH distance, and the smaller the energy consumption for node-to-CH communications (Fig. 4). On the other hand, CHs communicate with the BS in single hop. Thus, the larger the number of CHs, the more nodes will be communicating single-hop with the BS, and the more energy will be spent. Taking both reasonings into account, one can find an optimal value for h, which minimizes the total energy consumption, and maximize the network's lifetime.

In SecLEACH, only a fraction of h CHs is probabilistically accessible (as determined by key sharing) by an ordinary node. That is, h is actually a *nominal* value; what ultimately matters is the *effective* value, h_e, given by $h_e = h \times P_s$. Note that, to obtain a given h_e, one does not need to start with a fixed h. In fact, one can first fix a value for P_s, and adjust h accordingly.

P_s and h will also determine the expected orphan rate, that is, the probability that an ordinary node will be orphan. Given P_s (and consequently $P_{\bar{s}}$) and h, the expected orphan rate P_o is given by $P_o = (P_{\bar{s}})^h$. In a network with n nodes, it is then expected that $n \times P_o$ nodes will be orphans, and communicating single-hop with the BS. Fig. 5 shows P_o as function of h under a $sl = 0.99$. Because P_o depends of the *absolute* number of the CHs, no matter the network size n, the amount of orphan nodes will be negligible for $h \leq 7$.

To show some concrete numbers and the tradeoffs induced by different parameter values, we provide estimates on energy consumption levels for different scenarios. For our estimates, we assume a network as in LEACH original paper, i.e., $n = 100$ nodes, uniformly distributed at random in a $10^4 m^2$ square area; and a BS located at the center of the square. We consider three key ring sizes for a fixed sl value ($m = 50, 100, 150$, for $sl = 0.99$) and three security levels for a fixed m value (*sl* $= 0.95, 0.98, 0.99$, for $m = 100$). Table 1 exhibts the respective P_s values for these scenarios. In each case, we take into account only the energy consumed for communication in the steady-state phase. Because SecLEACH adds just a few extra bytes in the setup phase communication ($\mid$ *seed*$_H$ + *nonce* $\mid$ and $\mid$ *id*$_K$ + $\mathsf{mac}_{k[r]}(msg)$ − *crc* $\mid$ bytes, Fig. 3, Step 1 and 2, respectively), we expected this overhead will be "broke up" along the cycles of the subsequent steady-state phase. In addition, we do not consider the cost of cryptographic operations, as it has been shown [31] that those employed in our solution incur.

scenario	**LEACH**	**sl=0.95**	**sl=0.98**	**sl=0.99**	**m=50**	**m=100**	**m=150**
P_s	1.0	0.995	0.87	0.636	0.396	0.636	0.78

Table 1: Energy Overhead

To estimate the energy consumption, we assume the same radio energy model used in LEACH [25]. In this model, a radio dissipates ϵ_r = 50 $\mathrm{nJ/bit}$ to run the transmitter or receiver circuitry, and ϵ_a = 100 $\mathrm{pJ/bit/m^2}$ for the transmitter amplifier. Also, the radios expend the minimum required energy to reach the recipients and are turned off to avoid receiving unintended transmissions. An d^2 energy loss due to channel transmission is assumed as well. Under this model, the costs to transmit ($\mathcal{E}_T$) and receive ($\mathcal{E}_R$) a β-bit message at distance d, and the amount of energy $\mathcal{E}_{cycle}$ the network consumes to go through one cycle of sensor data reporting are given respectively by:

$$\begin{aligned} \mathcal{E}_T(\beta, d) &= \beta\,\epsilon_r + \beta\,d^2\,\epsilon_a \\ \mathcal{E}_R(\beta) &= \beta\,\epsilon_r \\ \mathcal{E}_{cycle} &= (n-h)\,[\mathcal{E}_T\,(\beta, d_1) + \mathcal{E}_R\,(\beta)] + h\,\mathcal{E}_T\,(\beta, d_2) \end{aligned}$$

where d_1 is average distance between an ordinary node and its closest CH, and d_2 is the average distance between a CH and the BS.

In what follows, we calculated d_1 and d_2 by using the derivation in Appendix A. Also, we set SecLEACH messages to be 36 bytes long (the maximum allowed in TinyOS [33]) and LEACH

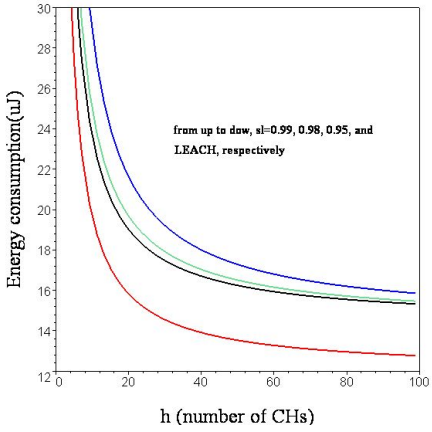

Figure 6: Average node-to-CH Distance, for m=100

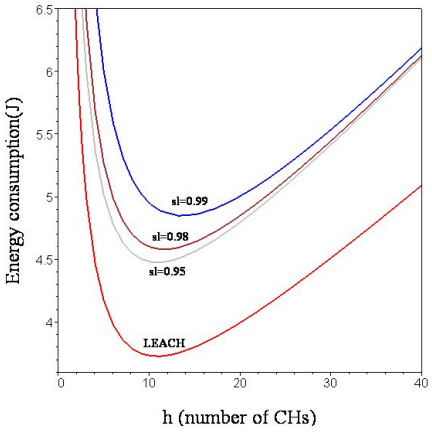

Figure 7: Energy consumption, for m=100

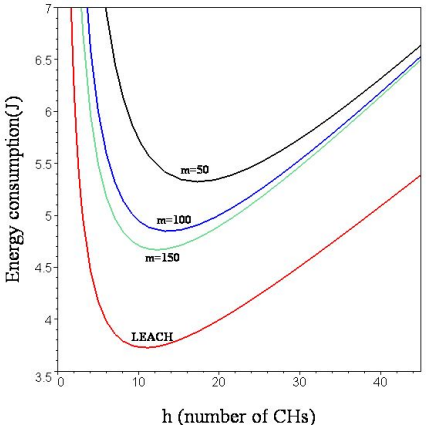

Figure 8: Energy consumption, for sl=0.99

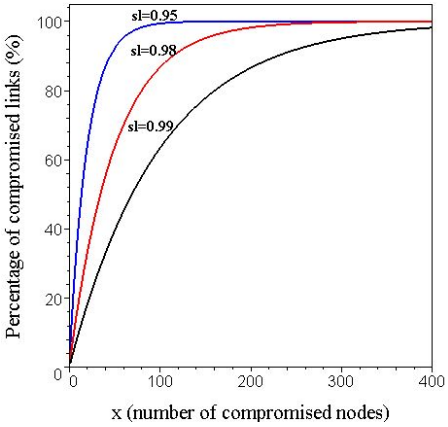

Figure 9: SecLEACH's resiliency against node capture

messages to be 30 bytes long. The difference is meant to account for the size difference between the MAC (8 bytes [31]) and CRC (2 bytes [34]) – the former present in SecLEACH, but absent in LEACH; and the latter present in LEACH, but absent in SecLEACH, as shown in Fig. 3, Steps 4 and 5.

Using the average distances and energy consumption numbers above, we obtained the energy consumption level for the various scenarios we considered. Figs. 7 shows the energy consumption in node-CH communication for different security levels. Note that the energy consumption is smaller in LEACH than in any instantiations of SecLEACH. Also, larger values of sl lead to larger overheads. On the other hand, the higher the h, the smaller the overhead.

Figs. 7 and 8 show the energy consumption level for one cycle of sensor data reporting in the steady-state phase, for the various scenarios we considered. Note that, in all cases, there is a value of h for which the energy consumption is minimum. Also, increases in the security level lead to increases in the energy consumption level (Fig. 7); and, for a given security level, larger key rings decrease the energy consumption (Fig. 8).

We also estimated how scalable SecLEACH is. Table 2 shows the overhead incurred by SecLEACH, under the various parameter values and under different network sizes n, as compared to LEACH. The overheads were computed using the values of h for which the energy consumption, in each scenario, is minimum. First of all, it is worth mentioning that overhead in SecLEACH is due to two factors: the increased msg size (20% larger) and the increased node-CH distance – the distance CH-BS is not increased, as every CH shares a key with the BS. Note that, for the maximum security level ($sl = 0.99$), the overhead increases from 30,0% to 46,1%, as the network

becomes larger. On the other hand, this overhead may be mitigated by using a larger m value, as shows the column $m = 150$. Alternatively, one may also choose to live with a lower security level. Because $sl = 0.95$ has a P_s very close to LEACH (Table 1), e.g., its overhead is mainly due to the increased msg size and, in turn, is very scalable.

n	sl=0.95	sl=0.98	sl=0.99	m=50	m=100	m=150
100	20,1%	22,9%	30,0%	42,9%	30,1%	25,3%
1000	20,2%	25,6%	39,8%	65,6%	39,8%	30,3%
10000	20,3%	27,4%	46,1%	80,6%	46,1%	33,6%

Table 2: Energy Overhead

5.2. Resiliency against node capture

In key distribution schemes, resiliency against node capture measures how much of the network (its communication links) is compromised when a node is compromised. It is a critical performance measure that gauges the robustness of a solution. In SecLEACH, the values of m and S determines the probability that a random link will be compromised when a node (that is not either end of the link) is compromised.

Fig. 9 shows the percentage P_c of compromised links as a function of the absolute number of compromised nodes for the considered security levels. (This figure was first shown in [7]). Note that P_c increases as the absolute number of compromised nodes (instead of percentage of nodes in the network) increases. Beyond a certain value of x, P_c reaches the value 1.0, no matter what is the security level. For smaller values of x, however, there is a significant difference in the P_c with different values of *sl*. This difference decreases as x increases.

6. Related Work

The number of studies specifically targeted to security of resource-constrained WSNs has grown significantly. Due to space constraints, we provide a sample of studies based on cryptographic methods, and focus on those targeted to clustered networks.

Perrig et al. [31] proposed a suite of efficient symmetric key based security building blocks. Eschenauer et al. [6] looked at random key predistribution schemes (which we discussed in Section 4.1), and originated a large number of follow-on studies [7,9,12,14,16,17]. Most of the proposed key distribution schemes, probabilistic or otherwise (e.g., [8]), are not tied to particular network organizations, although they mostly assume flat network, with multi-hop communication. Thus they are not well suited to clustered networks. Still others (e.g., [35, 36]) focused on detecting and dealing with injection of bogus data into the network.

Among those specifically targeted to cluster-based sensor networks, Bohge et al. [37] proposed an authentication framework for a concrete 2-tier network organization, in which a middle tier of more powerful nodes between the BS and the ordinary sensors were introduced for the purpose of carrying out authentication functions. In their solution, only the sensor nodes in the lowest tier do not perform public key operations. More recently, Oliveira et al. [38] propose solution that relies exclusively on symmetric key schemes and is suitable for networks with an arbitrary number of levels; and Ferreira et al. [32] proposed SLEACH, which we discussed in Section 3.

7. Conclusion

In this paper, we presented SecLEACH, a protocol for securing node-to-node communication in LEACH-based networks. SecLEACH bootstraps its security from random key predistribution, and can yield different performance numbers on efficiency and security depending on its various parameter values. Our estimates show that the overhead incurred by SecLEACH is manageable; and memory usage, energy efficiency, and security level can be each traded off for another, depending on what is most critical in a system.

References

[1] Deborah Estrin, Ramesh Govindan, John S. Heidemann, and Satish Kumar. Next century challenges: Scalable coordination in sensor networks. In *Mobile Computing and Networking*, pages 263–270, Seattle, WA USA, 1999.

[2] G. J. Pottie and W. J. Kaiser. Wireless integrated network sensors. *Commun. ACM*, 43(5):51–58, 2000.

[3] C. Karlof and D. Wagner. Secure routing in wireless sensor networks: Attacks and countermeasures. *Elsevier's AdHoc Networks Journal, Special Issue on Sensor Network Applications and Protocols*, 1(2–3):293–315, September 2003.

[4] A. D. Wood and J. A. Stankovic. Denial of service in sensor networks. *IEEE Computer*, 35(10):54–62, October 2002.

[5] D. W. Carman, P. S. Kruus, and B. J. Matt. Constraints and approaches for distributed sensor network security. Technical report, NAI Labs, The Security Research Division, Network Associates, Inc., http://www.nai.com/research/nailabs/cryptographic/a-communications-security.asp, 2000.

[6] L. Eschenauer and V. D. Gligor. A key management scheme for distributed sensor networks. In *Proceedings of the 9th ACM conference on Computer and communications security*, pages 41–47. ACM Press, 2002.

[7] H. Chan, A. Perrig, and D. Song. Random key predistribution schemes for sensor networks. In *IEEE Symposium on Security and Privacy*, page 197. IEEE Computer Society, may 2003.

[8] S. Zhu, S. Setia, and S. Jajodia. Leap: efficient security mechanisms for large-scale distributed sensor networks. In *Proceedings of the 10th ACM conference on Computer and communication security*, pages 62–72. ACM Press, 2003.

[9] D. Liu and P. Ning. Establishing pairwise keys in distributed sensor networks. In *10th ACM Conference on Computer and Communications Security (CCS '03)*, pages 52–61, 2003.

[10] D. Liu and P.Ning. Efficient distribution of key chain commitments for broadcast authentication in distributed sensor networks. In *10th Annual Network and Distributed Systems Security Symposium (NDSS'03)*, pages 263–276, 2003.

[11] D. Liu and P. Ning. Location-based pairwise key establishments for static sensor networks. In *1st ACM workshop on Security of ad hoc and sensor networks (SASN´03)*, pages 72–82. ACM Press, 2003.

[12] W. Du, J. Deng, Y. S. Han, and P. K. Varshney. A pairwise key pre-distribution scheme for wireless sensor networks. In *10th ACM conference on Computer and communication security (CCS´03)*, pages 42–51. ACM Press, October 2003.

[13] Q. Huang, J. Cukier, H. Kobayashi, B. Liu, and J. Zhang. Fast authenticated key establishment protocols for self-organizing sensor networks. In *2nd ACM international conference on Wireless sensor networks and applications (WSNA'03)*, pages 141–150. ACM Press, 2003.

[14] W. Du, J. Deng, Y. S. Han, S. Chen, and P. Varshney. A key management scheme for wireless sensor networks using deployment knowledge. In *IEEE INFOCOM'04,*, 2004.

[15] D. Huang, M. Mehta, D. Medhi, and L. Harn. Location-aware key management scheme for wireless sensor networks. In *2nd ACM workshop on Security of ad hoc and sensor networks (SASN'04)*, pages 29–42. ACM Press, 2004.

[16] S. Zhu, S. Xu, S. Setia, and S. Jajodia. Establishing pairwise keys for secure communication in ad hoc networks: A probabilistic approach. In *11th IEEE International Conference on Network Protocols (ICNP'03)*, page 326, Atlanta, USA, November 2003. IEEE Computer Society.

[17] R. Kannan, L. Ray, and A. Durresi. Efficient key pre-distribution schemes for sensor networks. In *1st European Workshop on Security in Wireless and Ad-Hoc Sensor Networks (ESAS´ 04)*, Heidelberg, Germany, August 2004.

[18] J. Hwang and Y. Kim. Revisiting random key pre-distribution schemes for wireless sensor networks. In *2nd ACM workshop on Security of ad hoc and sensor networks*, pages 43–52. ACM Press, 2004.

[19] *Combinatorial Design of Key Distribution Mechanisms for Wireless Sensor Networks.*, Sophia Antipolis, France, September 2004. Lecture Notes in Computer Science.

[20] J. L. Hill and D. E. Culler. Mica: A wireless platform for deeply embedded networks. *IEEE Micro*, 22(6):12–24, 2002.

[21] Crossbow Technology, Inc., 41 Daggett Dr., San Jose, CA 95134. *MPR/MIB Mote Hardware Users Manual – Document 7430-0021-05*, December 2003.

[22] Sameer Tilak, Nael B. Abu-Ghazaleh, and Wendi Heinzelman. A taxonomy of wireless micro-sensor network models. *ACM Mobile Computing and Communications Review*, 6(2):28–36, April 2002.

[23] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *IEEE Communications Magazine*, 40(8):102–114, August 2002.

[24] Edgar H., Jr. Callaway, and Edgar H. Callaway. *Wireless Sensor Networks: Architectures and Protocols*. CRC Press, hardcover edition, August 2003.

[25] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *IEEE Hawaii Int. Conf. on System Sciences*, pages 4–7, january 2000.

[26] A. Manjeshwar and D. Agrawal. Teen: A routing protocol for enhanced effeciency in wireless sensor networks. In *1st International Workshop on Parallel and Distributed Computing Issues in Wireless Networks and Mobile Computing*, 2001.

[27] O. Younis and S. Fahmy. Distributed clustering in ad-hoc sensor networks: A hybrid, energy-efficient approach," ieee infocom, march 2004. In *IEEE INFOCOM*, pages 629–640, March 2004.

[28] Qing Fang, Feng Zhao, and Leonidas Guibas. Lightweight sensing and communication protocols for target enumeration and aggregation. In *4th ACM international symposium on Mobile ad hoc networking & computing (MOBICHOC´ 03)*, pages 165–176. ACM Press, 2003.

[29] A. Iwata, C. Chiang, G. Pei, M. Gerla, and T. Chen. Scalable routing strategies for ad-hoc wireless networks. *IEEE Journal on Selected Areas in Communications*, 17(8):1369–1379, Aug. 1999. Special Issue on Ad-Hoc Networks.

[30] B. Schneier. *Applied Cryptography*. Wiley, second edition, 1996.

[31] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J. D. Tygar. SPINS: Security protocols for sensor networks. *Wireless Networks*, 8(5):521–534, September 2002.

[32] A. C. Ferreira, M. A. Vilaça, L. B. Oliveira, E. Habib, H. C. Wong, and A. A. Loureiro. On the security of cluster-based communication protocols for wireless sensor networks. In *4th IEEE International Conference on Networking (ICN'05)*, volume 3420 of *Lecture Notes in Computer Science*, pages 449–458, Reunion Island, April 2005. Springer.

[33] Jason Hill, Robert Szewczyk, Alec Woo, Seth Hollar, David Culler, and Kristofer Pister. System architecture directions for networked sensors. In *Proceedings of the ninth international conference on Architectural support for programming languages and operating systems*, pages 93–104. ACM Press, 2000.

[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 (SensSys 2004)*, November 2004.

[35] B. Przydatek, D. Song, and A. Perrig. SIA: Secure information aggregation in sensor networks. In *ACM SenSys 2003*, Nov 2003.

[36] F. Yea, H. Luo, S. Lu, and L. Zhang. Statistical en-route filtering of injected false data in sensor networks. In *INFOCOM 2004*, 2004.

[37] M. Bohge and W. Trappe. An authentication framework for hierarchical ad hoc sensor networks. In *Proceedings of the 2003 ACM workshop on Wireless security*, pages 79–87. ACM Press, 2003.

[38] Leonardo B. Oliveira, Hao Chi Wong, and Antonio A. F. Loureiro. Lha-sp: Secure protocols for hierarchical wireless sensor networks. In *9th IFIP/IEEE International Symposium on Integrated Network Management (IM'05)*, pages 31–44, Nice, France, May 2005.

[39] Philip J. Clark and Francis C. Evans. Distance to nearest neighbor as a measure of spatial relationships in populations. *Ecology*, 35(4):445–53, 1954.

[40] K. P. Donnelly. Simulations to determine the variance and edge effect of total nearest neighbour distance (ed. by i. hodder), 1978. Simulation Studies in Archaeology.

A. Average Distance Estimates

Next, we will present the derivation for the distance estimates. It is worth noting that both are independent of the network size n.

A.1. Distance between CH and BS

Given a square of side length $2\,s$ the probability P that the distance of a randomly chosen point in the square to its center is less or equal to x is given by:

$$P(d \leq x) = \frac{\pi\, x^2}{4\, s^2},\ \text{if } 0 \leq x \leq s$$
$$= \frac{\pi\, x^2 - 4\left(x^2 \arctan\left(\frac{\sqrt{x^2 - s^2}}{s}\right) - s\sqrt{x^2 - s^2}\right)}{4\, s^2},$$
$$\text{if } s \leq x \leq \sqrt{2}\, s$$

Hence, this probability density function (pdf) has the form:

$$f(x) = \frac{\pi\, x}{2\, s^2}, \qquad \text{if } 0 \leq x \leq s$$
$$= \frac{\pi\, x}{2\, s^2} - \frac{2\, x\, arctan\left(\frac{\sqrt{x^2 - s^2}}{s}\right)}{s^2}, \qquad \text{if } s \leq x \leq \sqrt{2}\, s$$

Now, we may use the pdf to calculate the expected distance:

$$E(X) = \int_0^{\sqrt{2}\, s} x\, f(x) = \frac{\left(\sqrt{2} + \log\left(1 + \sqrt{2}\right)\right)\ s}{3}$$

A.2. Distance between ordinary node and CH

In a population of i individuals distributed at random in an area a, the expected distance from an individual to its nearest neighbor (NN) [39], and the same with edge effect correction [40] (NN_c), are, respectively, given by:

$$NN = 0.5\sqrt{\frac{1}{\rho}}$$
$$NN_c = 0.5\sqrt{\frac{a}{i}} + \left(0.0514 + \frac{0.041}{\sqrt{i}}\right)\frac{p}{i}$$

where ρ stands for neighborhood density, i.e., $\rho = \frac{i}{a}$ and p is the perimeter of the study area a [1].

To determine the expected distance from an ordinary node to its nearest CH, we may consider only the CHs as neighbors of this node and apply the formula for NN calculation. Thus, this expected distance for LEACH and SecLEACH are given, respectively, by:

$$NN_{LEACH} = 0.5\sqrt{\frac{a}{h+1}} + \frac{\left(0.0514 + \frac{0.041}{\sqrt{h+1}}\right)\ p}{h+1}$$
$$NN_{SecLEACH} = 0.5\sqrt{\frac{a}{h_e+1}} + \frac{\left(0.0514 + \frac{0.041}{\sqrt{h_e+1}}\right)\ p}{h_e+1}$$

[1] Individuals have been treated as dimensionless points by Clark and Evans, since their dimensions are usually negligible as compared to the total area